UNIVERSIDAD DE GUAYAQUIL

Facultad de Ciencias Matemáticas y Físicas

Carrera de Ingeniería en Sistemas Computacionales

"Desarrollo del Módulo de Ventas Operacionales para un E-CRM"

TESIS DE GRADO

Previo a la Obtención del Título de:

INGENIERO EN SISTEMAS COMPUTACIONALES

Autor(es):

SILVANA CAROLINA PAREDES VANEGAS

MIGUEL SALOMÓN CARVAJAL DÁVALOS

VANESSA MARGARITA GARCÍA PICO

GUAYAQUIL-ECUADOR

Año: 2006 – 2007

AGRADECIMIENTO

A mis padres, por su apoyo incondicional; a las personas que aportaron con sus ideas, y a todos quienes colaboraron para llevar a cabo este proyecto.

Silvana Paredes

DEDICATORIA

Dedico esta obra a mis padres, quienes han sido mis principales maestros desde la infancia.

Silvana Paredes

AGRADECIMIENTO

A mi Sra. Madre: María Luisa Dávalos Gómez

A mi Sr. Padre: Miguel Salomón Carvajal Becerra

Divinos seres que me dieron la vida y grandes maestros que tengo de toda la vida y gracias a su apoyo y coraje, forjaron mi carácter y espíritu de salir adelante.

Miguel S. Carvajal Dávalos

DEDICATORIA

El esfuerzo entregado en esta obra le ofrezco a Dios, cuya bendición me ha permitido conseguir este logro, el mismo que lo dedico a mis padres, hermanos, maestros, amigos, compañeros y demás personas que creyeron en mis habilidades y aportaron su granito de arena para conseguir este triunfo.

Miguel S. Carvajal Dávalos

AGRADECIMIENTO

Agradezco a Dios que su infinita bondad y amor me permite cumplir mis metas. A mis padres por su apoyo incondicional, esfuerzo y confianza que han puesto en mi. A todos por quiénes ha sido posible obtener mi título universitario. A todos Gracias!

Vanessa García Pico

DEDICATORIA

A mis padres por su apoyo, comprensión y paciencia.
Que Dios los bendiga hoy y siempre.

Vanessa García Pico

TRIBUNAL DE GRADUACIÓN

_____ _____

_____ _____

_____

DECLARACIÓN EXPRESA

“La autoría de la tesis de grado corresponde exclusivamente al suscrito(s), perteneciendo a la Universidad de Guayaquil los derechos que generen la aplicación de la misma”

(Reglamento de Graduación de la Carrera de Ingeniería en sistemas Computacionales, Art. 26)

RESUMEN

El Módulo de Ventas Operacionales permite realizar las operaciones involucradas en el proceso de venta: recepción de solicitudes de clientes, emisión de facturas, establecimiento de contrato entre el cliente y la empresa, el otorgamiento o denegación de créditos a los clientes de acuerdo con su historial, y el registro de los pagos que deben realizar los clientes en el período establecido.

Permite administrar al personal del área de ventas, asignar las comisiones de acuerdo con las políticas de la empresa, emitir reportes mensuales para su respectivo pago. Dentro de las actividades diarias de los asesores de ventas se maneja la asignación del cronograma de visitas que deben realizar a los clientes VIP.

La aplicación de un CRM para una empresa de servicios mejora la atención que se brinda a los clientes, tanto en calidad como en tiempo, debido a que se ofrecen varias vías por medio de las cuales puedan solicitar sus servicios requeridos; pueden realizarlo por medio del Call Center, sitio web de la empresa, o mediante visitas efectuadas en el caso de los clientes VIP.

De esta manera, se pretende crear un vínculo a largo plazo con el cliente, garantizando su lealtad al enfocarse en las necesidades que presenten.

INDICE GENERAL

MANUAL TÉCNICO

1. Introducción 2

1.1 Herramientas utilizadas 3

1.1 1 Sun Java Studio Creator:Requisitos 4

1.2 Base de datos 7

1.2.1 Estándares de diseño de base de datos 8

1.2.1.1. Tablas 8

1.2.1.2 Campos. 9

1.2.1.3. Indíces 15

1.2.1.4. Vistas 15

1.2.1.5. Secuencias 16

1.2.1.6. Stored Program Units 16

1.2.1.7. Variables 17

1.2.1.8. Parámetros de entrada y salida 17

1.3 Estándares utilizados en la aplicación 19

1.4 Implementación de código 21

1.4.1 Código de construcción del menú 22

1.4.2 Método prerender 22

1.4.3 Método destroy 22

1.4.4 Clase conexión 23

1.4.5 Código para los botones de selección 24

1.4.6 Código para los controles calendario 25

1.4.7 Método valida_graba 26

1.4.8 Código del método manejador del evento Action 28

1.4.9 Código para importar un archivo plano 29

1.4.10 Clase manejadora de los datos de canal 31

1.4.11 Código para imprimir reportes 32

1.4.12 Clase Valida Números 33

1.5 Diccionario de datos 36

1.6 Diagrama de clases 61

1.7 Diagrama entidad-relación 62

1.8 PL/SQL 63

1.8.1. Funciones almacenadas 63

1.8.1.1.función vop_f_insertar_cab_solicitud 63

1.8.1.2. función vop_f_insertar_oferta 65

1.8.1.3. función vop_f_verificar_cedula_ruc 66

1.8.1.4. función vop_f_verificar_fecha_nac 68

1.8.2. Procedimientos almacenados 69

1.8.2.1. Procedimiento vop_p_actualizar_canal 69

1.8.2.2. Procedimiento vop_p_actualizar_ciclo 70

1.8.2.3. Procedimiento vop_p_actualizar_oferta 71

1.8.2.4. Procedimiento vop_p_actualizar_prospecto 72

1.8.2.5. Procedimiento vop_p_actualizar_rango_vtas 73

1.8.2.6. Procedimiento vop_p_actualizar_telf_vend 74

1.8.2.7. Procedimiento vop_p_actualizar_tipo_vendedor 75

1.8.2.8. Procedimiento vop_p_actualizar_vendedor 76

1.8.2.9. Procedimiento vop_p_actualizar_oferta_serv 77

1.8.2.10.Procedimiento vop_p_actualizar_visita 78

1.8.2.11.Procedimiento vop_p_eliminar_canal 78

1.8.2.12.Procedimiento vop_p_ eliminar _ciclo 79

1.8.2.13.Procedimiento vop_p_ eliminar _comi_cod_vend 79

1.8.2.14.Procedimiento vop_p_ eliminar _oferta 80

1.8.2.15.Procedimiento vop_p_ eliminar _prospecto 80

1.8.2.16. Procedimiento vop_p_ eliminar _rango 81

1.8.2.17. Procedimiento vop_p_ eliminar _telf_codvend 81

1.8.2.18. Procedimiento vop_p_ eliminar _tipo_vend 82

1.8.2.19. Procedimiento vop_p_ eliminar _vendedor 82

1.8.2.20. Procedimiento vop_p_ eliminar _visit_codvend 83

1.8.2.21. Procedimiento vop_p_ eliminar _oferta_serc 83

1.8.2.22. Procedimiento vop_p_ insertar_canal 84

1.8.2.23. Procedimiento vop_p_ insertar _ciclo 85

1.8.2.24. Procedimiento vop_p_ insertar _cliente 86

1.8.2.25. Procedimiento vop_p_ insertar_det_solicitud 89

1.8.2.26. Procedimiento vop_p_ insertar _prospecto 90

1.8.2.27. Procedimiento vop_p_ insertar _rango_ventas 91

1.8.2.28. Procedimiento vop_p_ insertar _telf_vend 92

1.8.2.29. Procedimiento vop_p_ insertar _tipo_vendedor 93

1.8.2.30. Procedimiento vop_p_ insertar _vendedor 95

1.8.2.32. Procedimiento vop_p_ insertar _visitas 96

1.8.2.33. Procedimiento vop_p_ registrar_pago 98

1.8.2.34. Procedimiento vop_p_ insertar _oferta_serv 99

1.8.3 Paquetes almacenados 100

1.8.3.1 Paquete para cálculo de comisiones 100

1.8.3.2 Paquete para facturación 118

1.8.3.3 Paquete para registrar contratos 134

1.9 Rowsets utilizados 150

MANUAL TÉCNICO

1. Introducción

El presente manual ha sido elaborado con el propósito de facilitar al programador las tareas de mantenimiento del módulo así como el control de cambios y versiones.

Las explicaciones dadas abarcan todo el desarrollo de la programación, comenzando por las herramientas recomendadas, la codificación de la aplicación realizada en Java y en PL/SQL, y los diagramas útiles para una mejor comprensión visual y global concerniente a este módulo.

Con la finalidad de simplificar la lectura del código se han aplicado estándares que deben ser estrictamente aplicados en los cambios posteriores que se realicen, y procurar en lo posible que no sean modificados los nombres de los elementos de base de datos presentados en esta versión, porque pueden ocasionar fallos en las aplicaciones que lo requieran.

1.1 Herramientas utilizadas en el módulo de ventas operacionales

Para el desarrollo de este módulo, se requieren herramientas que permitan modelar la capas de datos, negocio y presentación, tal como se describe a continuación:

- **Sun Java Studio Creator:** Capa de presentación

  Desarrollo de la interfaz gráfica en ambiente web y clases encargadas de manipular los datos que deben ser enviados y obtenidos de la base de datos.

- **PL/SQL:** Capa de negocio

  Desarrollo de paquetería, funciones y procedimientos almacenados que interactúan con la base de datos.

- **Oracle 9i:** Capa de datos

  Creación de usuarios, tablas, vistas, índices.

1.1.1. Sun Java Studio Creator: Requisitos del sistema

Sistema Operativo Solaris (SPARC Platform)

Componentes	Requerimientos
CPU	UltraSPARC III at 750 MHz velocidad de procesamiento
RAM	1 GB
Espacio en disco	650 MB (1Gb adicional requerido para la instalación)
Sistema Operativo	Solaris 9 4/04 OS Solaris 10 OS (Incluido el más reciente parche J2SE para Sun)
Navegador de Soporte	Sun soporta Mozilla 1.7

Sistema Operativo Solaris (x86 Platform Edition)

Componentes	Requerimientos
CPU	Intel Pentium 4 (o equivalente) a 1 GHz velocidad procesamiento
RAM	1 GB
Espacio en disco	750 MB (1Gb adicional requerido para la instalación)
Sistema Operativo	Solaris 9 4/04 OS Solaris 10 OS (Incluido el más reciente parche J2SE para Sun)
Navegador de Soporte	Sun soporta Mozilla 1.7

Plataforma Microsoft Windows

Componentes	Requerimientos
CPU	Intel Pentium 4 (o equivalente) a 1 Ghz de velocidad de procesamiento
RAM	1 GB
Espacio en disco	650 MB (300Mb adicional requerido para la instalación)
Sistema Operativo	Windows 2000 Professional Edition (Service Pack 4) Windows XP Professional and Home Editions (Service Pack 2) Windows Server 2003
Navegador de Soporte	Internet Explorer 5.5 Service Pack 2 (Windows 2000) Internet Explorer 6 Service Pack 2 (Windows XP)

Plataforma Linux

Componentes	Requerimientos
CPU	Intel Pentium 4 (o equivalente) a 1 Ghz de velocidad de procesamiento
RAM	1 GB
Espacio en disco	750 MB (300Mb requerido para la instalación)
Sistema Operativo	Red Hat Enterprise Linux 3
Navegador de Soporte	At least Firefox 1.0.4

Apple Macintosh Platform

Componentes	Requerimientos
CPU	1 Ghz PowerPC G4 1.83 GHz Intel Dual Core
RAM	1 Gbyte
Espacio en disco	450 MB (300Mb requerido para la instalación)
Sistema Operativo	Mac OS X 10.3.4, 10.4 Mac OS X 10.4.5
Máquina Virtual de Java	1.4.2_05 for 10.3 1.4.2_09 for 10.4 1.5.0_05 for 10.4.5
Navegador de Soporte	Safari 1.2.1, Mozilla 1.6 Safari 2 minimo for OS 10.4.5

Características y beneficios

Sun Java Studio Creator es un ambiente de desarrollo web visual fácil de utilizar para la plataforma Java. Permite crear visualmente aplicaciones web basadas en los estándares y portlets utilizando JavaServer Faces. La integración con fuentes de datos heterogéneas como bases de datos relacionales, JavaBeans empresariales y servicios web es simple. Muchas características muy convenientes también han sido incluidas.

Java Studio Creator 2 esta construido en NetBeans 4.1 e incluye características como un nuevo grupo de componentes JavaServer Faces components, soporte para crear portlets JSR-168, acceso simple a bases de datos, un modelo de aplicación streamlined y muchas cosas mas.

1.2 Base de datos

- La base de datos utilizada para este proyecto ha sido definida con el nombre de CRM.
- Se debe crear un usuario. En este caso, también se lo ha denominado CRM
- Se utilizan dos tablespaces. CRM_DAT se utiliza para las tablas y CRM_IDX para los índices.
- Una vez realizados todos los pasos anteriores, es posible ejecutar el script de creación de las tablas.

1.2.1 Estándares de diseño de base de datos

Dentro de estos estándares de diseño utilizaremos prefijos para creación de tablas, constraints, índices, vistas, secuencias, stored program units, variables y parámetros de entrada y salida.

1.2.1.1 Tablas

Los nombres de Tablas están conformado por:

Prefijo Nombre Sistema.

Está compuesto por tres caracteres que identifican al nombre del Sistema.
Ejemplo: CRM

Prefijo Nombre Módulo.

Está compuesto por tres caracteres que identifican al nombre del Módulo.
Ejemplo: VOP

Nombre de Tabla.

Está compuesto hasta 20 caracteres que identifican al nombre de la tabla.
Ejemplo: CANAL

Para establecer nombres largos utilizar abreviaturas de hasta 5 caracteres máximo por cada palabra y un total de hasta 4 palabras o nemónicos, estableciendo una sumatoria tope de 30 caracteres por todo el nombre de la tabla.
Así mismo haremos uso del caracter especial "_" UNDER SCORE para separar palabras o nemónicos en un nombre de tabla u otros objetos.

Ejemplo: CRM_VOP_CAB_FACTURA

1.2.1.2 Campos.

El campo especifica el nombre de una columna de la tabla. El nombre del campo debe estar conformado de la siguiente manera:

Sintaxis: PrefijoNombreTabla_ NombredelCampo

Ejemplo: CNL_CODIGO

1.2.1.3 Constraints.

Los constraints pueden ser definidos por diferentes tipos y dependiendo del tipo de restricción que se desee utilizar. Constraint es una restricción en los valores que puede tomar una o varias columnas de una tabla. Existiendo los siguientes tipos de constraints:

Constraints de Integridad:

- ♠ Not Null Constraints.
- ♠ Check Constraints.
- ♠ Unique Key Constraints.
- ♠ Primary Key Constraints.

Constraints de Integridad Referencial o Relacional:

- ♠ Foreign Key Constraints.

Not Null Constraint

Los constraints not null pueden ser definidos en el momento de la creación de la tabla o por medio de un comando que altere la estructura de la misma. La convención de nombrado para este tipo de constraints deberá ser la concatenación de prefijo del módulo al que pertenece el objeto, más el prefijo de 2 caracteres del tipo de constraint utilizado “NN” y una descripción de la tabla y columna afectada o sobre la que actúa la restricción NOT NULL. En ambos caso de sentencias CREATE TABLE o ALTER TABLE no es necesario definir el nombre del constraint not null. Pero por unificación y estandarización se debe utilizar nombres apropiados para este tipo de constraints según el sgte estandart, sino Oracle creará un constarint de nombre SYS_XXXXX para estos constraints.

Ejemplo: VOP_NN_CIF_CODIGO

3 caracteres para el prefijo del modulo + separador “_”

2 caracteres para el prefijo del tipo de restricción (constraint) + separador “_”

4 o 5 caracteres para la descripción de la tabla + separador “_”

4 o 5 caracteres para la descripción de la columna

Check Constraint

Los constraints check o de chequeo explícitamente define una condición en los valores posibles de una columna en una tabla. Los valores que se indican en el chequeo del valor de la columna deben ser explícitos y no pueden contener :

- ♠ Consultas que hagan referencias a valores en otros registros de otras tablas.
- ♠ Llamadas a funciones SYSDATE, UID, USER, USERENV, etc.
- ♠ Constantes tipo fecha que no son totalmente especificadas.

Se pueden definir este tipo de constraints con las instrucciones CREATE TABLE o ALTER TABLE.

Ejemplo:

```
CREATE TABLE CRM_VOP_PROSPECTO (
        PRO_CODIGO_PROSPECTO  NUMBER CONSTRAINT
        VOP_NN_CLI_CODIGO_PROSPECTO NOT NULL,
        PRO_SEXO              VARCHAR2(1)  CONSTRAINT
        SCO_CK_PRO_ESTADO CHECK (PRO_SEXO IN('F','M'))
)
```

3 caracteres para el prefijo del modulo + separador “_”
2 caracteres para el prefijo del tipo de restricción (constraint) + separador “_”
4 o 5 caracteres para la descripción de la tabla + separador “_”
4 o 5 caracteres para la descripción de la columna

Unique Key Constraint.

El constraint UNIQUE designa una columna o una combinación de columnas como clave única. Para satisfacer un constraint UNIQUE, no deben existir 2 registros en la tabla que puedan tener el mismo valor para la clave única. Sin embargo, la clave única definida de una sola columna puede contener valores nulos.

Una columna con clave única no puede ser de los tipos de datos LONG o LONG RAW. Usted no podrá designar la misma columna o combinación de estas como clave única y a la vez como clave primaria. Sin embargo, si podrá designar la misma columna o combinación de columnas como clave única y clave foránea.

Se puede definir la clave única en una simple columna o combinación de ellas con la sintaxis de un constraint de columna, desde una sentrencia CREATE TABLE O ALTER TABLE.

Convención: ADM_UK_DEPT_LOC_CIU_AGE

3 caracteres para el prefijo del modulo + separador “_”
2 caracteres para el prefijo del tipo de restricción (constraint) + separador “_”
4 o 5 caracteres para la descripción de la tabla + separador “_”
3 o 4 caracteres para la descripción de las columnas, utilizando separador “_”

Primary Key Constraint.

El constraint PRIMARY KEY designa una columna o una combinación de columnas como el primary key o clave primaria de la tabla. Para satisfacer este constraint de PRIMARY KEY, las siguientes condiciones deberán ser verdaderas:

1.- No puede existir más de un registro en la tabla con los valores de la clave primaria. O sea, la clave primaria de un registro no la puede tener otro registro es única.

2.- La columna o combinación de columnas que conforman la clave primaria no deben tener nulos. Y estas columnas deben estar definidas como NOT NULL.

3.- Una tabla sólo puede tener una sola clave primaria.

Una clave primaria no puede ser de tipos de datos LONG or LONG RAW. No se puede designar la misma columna o combinación de estas como clave primaria y a la vez como clave única. Sin embargo, se puede designar la misma columna o combinación de columnas como clave primaria y clave foránea a la vez.
Podemos hacer uso de un constraint de columna en la sintaxis para definir un primary key.

Convención: VOP_PK_CANAL

3 caracteres para el prefijo del modulo + separador “_”
2 caracteres para el prefijo del tipo de restricción (constraint) + separador “_”
20 caracteres nombre de la tabla

Foreign Key Constraint (Restricciones de Integridad Referencial)

Una restricción de integridad referencial designa una columna o una combinación de columnas como clave foránea y establece una relación entre la clave foránea y la clave primaria o la clave única especificada, esta relación es llamada la clave referencial. En esta relación, la tabla que contiene la clave foránea es llamada la tabla HIJA y la tabla que contiene su clave primaria o única referenciada es llamada la tabla PADRE.

Para poder definir constraints de claves foráneas en Oracle se debe tener en cuenta:

1.- La tabla HIJA y la tabla PADRE deben estar en la misma base de datos.

2.- El valor de el registro que tiene en la clave foránea debe aparecer con un valor referenciado a la clave primaria del registro de la tabla PADRE. El registro en la tabla hija es dicho a depender en la clave referenciada en la tabla padre.

3.- El valor de una de las columnas que realiza la clave foránea puede ser nulo.

Una clave foránea no puede ser de tipos de datos LONG or LONG RAW. Ud. Puede designar la misma columna o combinación de columnas como un foreign key y un primary key o unique key.

Para el nombrado de las claves foráneas, se deberán anteponer prefijos del Módulo al que pertenece el objeto, otro prefijo es el tipo de constraint **“FK”** y el nemónico o descripción abreviada de la tabla HIJA y luego la tabla PADRE.

Convención: ADM_FK_VEND_CANAL

3 caracteres para el prefijo del modulo + separador “_”
2 caracteres para el prefijo del tipo de restricción (constraint) + separador “_”
10 caracteres para la descripción de la tabla HIJA (Ej. USUA de Usuarios)
10 caracteres para la descripción de la tabla PADRE (Ej. EMPL de Empleados)

1.2.1.3 Indices.

Los índices tienen antepuesto prefijos antes del nombre del mismo, otro perfijo adicional es la letra I (de Indice) y a continuación la descripción del mismo. Si es el caso de un Primary Key, Foreign Key, Unique Key el nombre del indice toma relación al nombre del constraint y sino al nombre del indice toma el de la tabla y columnas que van ser indizadas.
En el caso de foreign keys, en la que se establece una relación de referencia a un PK de una tabla padre, los nombres del índice deberán tomar el mismo estándar del Foreign key. Estos nombres deberán ser abreviados (máximo 30 caracteres incluyendo prefijos).
Recordar que el constraint de Primary key, crea automáticamente un índice único con igual nombre del constraint, si se omite el nombre del constraint toma un secuencial SYS_XXXXXX

1.2.1.4. Vistas.

Las vistas son tablas lógicas que tienen acceso a datos de otras tablas y/o vistas, están implementadas por una consulta donde se pueden mostrar datos de diferentes tablas . Para la convención de nombrado también poseen un prefijo que identifica al sistema, seguido del prefijo del módulo que pertenece, más la letra V (Vista) y por último el nombre de la vista.

Ejemplo: CRM_VOP_V_ROTACION

1.2.1.5 Secuencias.

Las secuencias son objetos de base de datos que normalmente son propios de un campo o columna Primary Key de tabla, estos también deben poseer el un prefijo que identifica al sistema, seguido del prefijo del módulo que pertenece, acompañado de una letra S (Secuencia) y al último nombre de la secuencia preferiblemente debe tener relación con la tabla y el campo o columna que almacenará la generación automática de este objeto.

Ejemplo: CRM_ADM_S_ACCESOS_CODIGO

1.2.1.6. Stored Program Units.

Son procedimientos, funciones o paquetes almacenados de base de datos que incluyen programación PL/SQL.

Para los Stored Program Units se utiliza la siguiente sintaxis:

PrefijoNombreMódulo_PrefijoStoredProgramUnits_NombreObjeto

Prefijo Nombre Módulo.

Esta compuesto por tres caracteres que identifican el nombre del Módulo.

Prefijo Stored Program Units.

Programas almacenados de base de datos.

Prefijo	Descripción
P	**Procedimiento (Procedure)**
F	**Función (Function)**
K	**Especificación del paquete (Package Specification)**
K	**Cuerpo del paquete (Package Body)**

Ejemplo:

ADM_P_ NOMBRE_STORED_PROCEDURE
ADM_F_ NOMBRE_STORED_FUNCTION
ADM_K_NOMBRE_STORED_PACKAGE_SPACIFICATION
ADM_K_ NOMBRE_STORED_PACKAGE_BODY

1.2.1.7. Variables.

Se puede declarar variables locales y globales en cualquier trigger, comando de item de menú, subprogramas de usuario ó PL/SQL que cree. Se puede usar cualquier item de forma para almacenar un valor como una variable.
Para definir de variables locales se antepone al nombre de la variable VL_ .
En las variables globales se antepone dos puntos y la palabra GLOBAL

1.2.1.8. Parámetros de Entrada y Salida.

Al declarar un parámetro debería indicar el tipo de dato y el modo de acceso dentro de la lista formal de parámetros. Los parámetros pueden ser declarados como tipos VARCHAR2, DATE, NUMBER ó BOOLEAN, ó cualquiera de los tipos de Objetos de Oracle Forms. Cuando usted cree una función o procedimiento ingrese los tipos de datos después del nombre de cada parámetro formal.

Ejemplo:

PROCEDURE VTA_P_DATOS_CLIENTE (VL_COD_CLI NUMBER)
FUNCTION VTA_F_SALDO_CREDITOS (VL_CLIENTE VARCHAR2) RETURN BOOLEAN

Los modos de acceso de un parámetro pueden ser IN (por defecto), OUT ö IN OUT. Un parámetro IN pasa un valor a un subprograma. El subprograma

puede leer el valor del parámetro de entrada (IN), pero no se puede escribir sobre este valor. Un parámetro de entrada actual puede ser una constante, variable inicializada o expresión.
Un parámetro de salida (OUT) retorna un valor desde un subprograma. Un parámetro actual de salida puede ser solo variable local y el programa asignaría un valor a la variable.
Un parámetro de entrada y salida (IN OUT) pasa un valor a un subprograma y retorna un valor desde el subprograma. A estas variables por lo tanto si se les puede asignar un valor pero no es requerido.

Ejemplo:
PROCEDURE VTA_P_DATOS_CLIENTE (VL_COD_CLI

Dentro de la definición de un procedimiento se puede utilizar variables de IN, OUT ö IN OUT, para el efecto citamos un ejemplo.

1.3 Estándares utilizados en la aplicación

Para construir la interfaz gráfica de la aplicación se utilizan los siguientes estándares en los controles utilizados:

Elemento	**Prefijo**	**Ejemplo**
Caja de texto	txt	txtDescripcion
Texto estático	txt	txtCodigo
Botón	btn	btnNuevo
Botón de selección	cmb	cmbVendedor
Casillas de verificación	chk	chkVendedor
Mensaje	mensaje	mensaje

De acuerdo con la funcionalidad que tiene cada página se clasifican en los siguientes:

Tipo	**Prefijo**	**Ejemplo**
Edición	edicion	edicionContratos
Mantenimiento	mantenimiento	mantenimientoCanal
Consultas	consulta	consultaComisiones
Procesos	proceso	procesoFacturacion

Los métodos de la clase DatosVop -que contiene las invocaciones a procedimientos, funciones y paquetes almacenados- se clasifican en tres tipos, y deben llevar el prefijo correspondiente a la funcionalidad dada.

Tipo de método	**Prefijo**	**Ejemplo**
Inserción	DML_Insertar	DML_InsertarCanal
Actualización	DML_Actualizar	DML_ActualizarCanal
Eliminación	DML_Eliminar	DML_EliminarCanal

1.4 Implementación de código

El módulo consta del siguiente tipo de clases:

Clase para establecer conexión a la base de datos

- Conexión

Clases que contienen invocaciones a procedimientos, funciones y paquetes de PL/SQL

- DatosVop
- FuncionesVop

Clases que contienen diseño de interfaz y gráfica

Todas las que empiecen con los siguientes prefijos:

- Mantenimiento
- Edición
- Consultas
- Procesos

Clases que manipulan tipos de datos similares a las tablas del sistema

Sus nombres se asemejan a los que definen a las tablas. Ej.: Canal.

Clase de validaciones de números

- ValidaNumeros

1.4.1 Código de construcción del menú

Este código está contenido en el método init de todas las páginas.

```
public void init() {
        super.init();
        try {
            _init();
        } catch (Exception e) {
            log("AdmjspPlantilla Initialization Failure", e);
            throw e instanceof FacesException ? (FacesException) e: new FacesException(e);
        }
        if(getVOPSessionBean1().isCrmSesion())
        {   // Limpiar y Crear Menú
            cConsultarMenu oCM = new cConsultarMenu();
            oCM.LimpiarArbol(treeMenu);
            treeMenu =
oCM.CargarMenu(treeMenu,getVOPSessionBean1().getCrmModulo(),getVOPSessionBean1().getCrmCod
Usu());
        }
    }
```

1.4.2 Método prerender

Se invoca al ejecutarse una página y luego de cada evento lanzado por un botón.

Utilizado para enviar valores a los rowsets que reciben parámetros.

```
public void prerender(){
        if ( txtCedula.getText() == null) {
            try{
                getVOPSessionBean1().getCrm_vop_cab_facturaRowSet().setInt(1, 0);
                crm_vop_cab_facturaDataProvider.refresh();
            }catch (Exception e) {
                info("Error");
            }
        }
    }
```

1.4.3 Método destroy

Utilizado por el IDE de Java para cerrar los DataProviders utilizados.

```
    public void destroy() {
        crm_vop_cab_facturaDataProvider.close();
    }
```

1.4.4 Código de clase Conexión

La clase conexión es utilzada por las clases DatosVop y FuncionesVop, para los accesos a la base de datos mediante los procedimientos, funciones y paquetes de PL/SQL.

```
package crm.vop;

import java.sql.*;
import javax.naming.*;
import javax.sql.*;

public class Conexion {

    public Connection getConexion()
    {
        Connection oCon=null;
        DataSource oDs;
        InitialContext ic;

        try {
            ic   = new InitialContext();
            oDs  = (DataSource)ic.lookup("jdbc/crm");
            oCon = oDs.getConnection();
        }
        catch(NamingException ne)  {
            ne.printStackTrace();
        }
        catch(SQLException se)  {
            se.printStackTrace();
        }
        return oCon;
    }

    public void cerrarConexion(Connection pCon)
    {
        try {
           pCon.close();
        }
        catch(SQLException se) {
            se.printStackTrace();
        }
    }
}
```

1.4.5 Código para los botones de selección

Las tablas creadas para visualización de datos, tienen en su primera columna botones de selección para tratar los datos de un registro seleccionado.

Se debe agregar el siguiente código en la página:

```
public String getFilaActual()
{
    return tableRowGroup1.getRowKey().getRowId();
}

private Object ultimoSeleccionado = "0";

public Object getRBSeleccionado()
{
   String sv = (String)radioButton1.getSelectedValue();
   return sv.equals(ultimoSeleccionado) ? sv : null;
}
public void setRBSeleccionado(Object seleccionado)
{
    if (seleccionado != null)
    {
        ultimoSeleccionado = seleccionado;
    }
 }
```

Y enlazar las siguientes propiedades con los valores indicados.

Propiedad	**Valor**
nombre	grupoBoton
selected	RBSeleccionado
selectedValue	FilaActual

1.4.6 Código para los controles calendario.

Al crear controles calendario para el manejo de fechas, se deben crear los siguientes métodos, en donde se definen los valores mínimos y máximos que mostrará el calendario:

```
private Date minCalDate;

public Date getMinCalDate() {
      java.util.Calendar date=
java.util.Calendar.getInstance(FacesContext.getCurrentInstance().getViewRoot().getLocale());
      date.set(java.util.Calendar.HOUR_OF_DAY, 0);
      date.set(java.util.Calendar.MINUTE, 0);
      date.set(java.util.Calendar.SECOND, 0);
      date.set(java.util.Calendar.MILLISECOND, 0);
      date.add(java.util.Calendar.YEAR, -100);
      return date.getTime();
}

private Date maxCalDate;

public Date getMaxCalDate() {
       java.util.Calendar date = GregorianCalendar.getInstance();
       date.set(java.util.Calendar.HOUR_OF_DAY, 0);
       date.add(java.util.Calendar.YEAR, 2);
       maxCalDate = date.getTime();
       return maxCalDate;
  }
```

Enlazar las propiedades del control como se indica a continuación:

Propiedad	**Valor**
minDate	minCalDate
maxDate	maxCalDate

Y colocar en el método prerender el siguiente código:

```
public void prerender() {
      minCalDate = getMinCalDate();
  }
```

1.4.7 Método valida_graba

Cada clase que realiza inserciones y actualizaciones contiene un método valida_graba que es invocado antes de realizar las dos operaciones de datos mencionadas. Si devuelve verdadero se considera que los datos son válidos y pueden ser grabados.

En este método se realizan las siguientes validaciones:

- Que se ingresen los datos obligatorios.
- Que cada control contenga el tipo de datos correspondiente.
- Que los controles descriptivos contengan menos de la longitud máxima permitida por la base de datos.
- Que la cédula o RUC sean válidos.
- Que se registren solo los clientes mayores de 18 años.

```
private boolean valida_graba(){
        boolean llexito=true;
        boolean bandCedula=true;
        boolean bandPasaporte=true;
        String resultado="";
        SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd");
        FuncionesVop oFunciones=new FuncionesVop();

        if (txtCedula.getText()==null || txtCedula.getText().equals("")){
            bandCedula = false;
        }

        if (txtPasaporte.getText()==null || txtPasaporte.getText().equals("")){
            bandPasaporte = false;
        }
        if (cmbEstadoCivil.getSelected() == null){
            info("No existe información de estado civil...");
            llexito=false;
        }
        if (cmbActividad.getSelected() == null){
            info("No existe información de actividades...");
            llexito=false;
        }
        if (cmbNacionalidad.getSelected() == null){
            info("No existe información de nacionalidad...");
            llexito=false;
        }
```

```
    if (tipo_persona.equals("1") && llexito){
        if (!bandCedula && !bandPasaporte){
            llexito=false;
            info("Debe ingresar Nº cédula o pasaporte");
        }else{
            if (bandCedula){
                if (!isNumberString(txtCedula.getText().toString())){
                    info("El Nº cédula/ruc solo debe contener números");
                    llexito=false;
                }

                if (llexito){
                    try{
                        resultado=new String(oFunciones.verificarCedula(txtCedula.getText().toString(), 1));
                    }catch(Exception e){
                        info("Error en la verificación de cedula...");
                        llexito=false;
                    }
                    if (resultado.equals("INCORRECTO")){
                        info("El Número de cédula o R.U.C. es incorrecto...");
                        llexito=false;
                    }
                }
            }

            if (bandPasaporte){
                if (txtPasaporte.getText().toString().length()>30){
                    info("Se han ingresado " + txtPasaporte.getText().toString().length() + " caracteres");
                    info("El pasaporte excede la longitud de 30 caracteres permitidos");
                    llexito=false;
                }
            }

            if (txtFechaNac.getText()==null || txtFechaNac.getText().equals("")){
                info("Debe registrar la fecha de nacimiento a esta persona natural...");
                llexito=false;
            }else{
                try{
                    resultado=new
String(oFunciones.verificarFechaNac(formatter.format(txtFechaNac.getSelectedDate())));
if (txtApellidos.getText().toString().length()>60){
            info("Se han ingresado " + txtCedula.getText().toString().length() + " caracteres");
            info("Los apellidos exceden la longitud de 60 caracteres permitidos");
            llexito=false;
        }
    }
    return llexito;
  }
```

1.4.8 Código del método manejador del evento Action de un botón

Este ejemplo muestra el código del botón Eliminar de la clase mantenimientoCanal.

Realiza la invocación al método DML_EliminarCanal de la clase DatosVop, para lo cual previamente se ha enviado el código de canal al método setCodigo de la clase Canal, debido a que DML_EliminarCanal obtiene los datos necesarios – en este caso el código de canal– desde la clase correspondiente. Finalmente este método que ha sido invocado devuelve un mensaje en el que indica si se pudo realizar la operación.

```
public String btnEliminar_action() {
    DatosVop oDatos = new DatosVop();
    oDatos.oCanal.setCodigo(txtCodigo.getText().toString());

   try{
      oDatos.DML_EliminarCanal();
      String mensaje = oDatos.DML_EliminarCanal();
      crm_vop_canalDataProvider.refresh();
      info(mensaje);

   } catch(SQLException se){
       info("Error al eliminar...");
   }
   btnEliminar.setDisabled(true);
   btnActualizar.setDisabled(true);
    return null;
}
```

La lógica utilizada en este método se aplica también para insertar y actualizar, con la diferencia que estos dos métodos invocan al método local valida_graba, y sólo si devuelve verdadero se manipulan los datos que serán ingresados a la base de datos.

1.4.9 Código para importar un archivo plano

Este método obtiene la ruta de un archivo agregado en el control Upload, verifica que sea un archivo de texto y que el tamaño no exceda de 1 MB. Luego lee el archivo y lo subdivide en tokens, es decir, en las partes que se encuentren entre el delimitador definido, en este caso: @.

Cada token corresponde a un campo, y se verifica que el tipo de datos sea correcto para finalmente insertar en la base de datos.

```
public String btnImportar_action() {
        //Obtener la ruta
        UploadedFile uploadedFile = (UploadedFile)fileUpload1.getUploadedFile();
        String texto = null;
        String ruta = uploadedFile.getOriginalName();
        String uploadedFileType = uploadedFileType = uploadedFile.getContentType();;
        Long uploadedFileSize = new Long(uploadedFile.getSize());
        boolean llExito = true;
        if (Integer.parseInt(uploadedFileSize.toString())/1024 > 100){
            llExito = false;
            info("El tamaño del archivo excede el máximo permitido");
        }
        if (!uploadedFileType.equals("text/plain")){
            llExito = false;
            info("Debe enviar un archivo de texto existente");
        }
        if (llExito){
                //Leer el archivo
                try{
                    BufferedReader entrada = new BufferedReader(new FileReader(ruta));
                    StringBuffer bufer = new StringBuffer();
                    while((texto = entrada.readLine()) != null){
                        bufer.append(texto);
                    }
                    texto = bufer.toString();
                }catch(IOException io){
                    info("Error");
                }
                 DatosVop oDatos = new DatosVop();
                String separador = "@";
                String numFactura = null;
                String codCliente = null;
                String fechaPago = null;
                int c=0;
                StringTokenizer st = new StringTokenizer(texto, separador);
                if (!st.hasMoreTokens()){
                    info ("El archivo no es válido");
                }
```

```
            //Obtener datos de los campos
            while (st.hasMoreTokens()) {
               c++;
               if (c == 1){
                  numFactura = st.nextToken();
                  if (isNumeric(numFactura)){
                        Datos.oCabFact.setNumeroFactura(Integer.parseInt(numFactura));
                  }else{
                     info("El archivo no es válido");
                     break;
                  }
               }
               if (c == 2){
                  codCliente = st.nextToken();
                  if (isNumeric(numFactura)){
                     oDatos.oCabFact.setCodCliente(Integer.parseInt(codCliente));
                  }else{
                     info("El archivo no es válido");
                     break;
                  }
               }
               if (c == 3){
                  fechaPago = st.nextToken();
                    String fecha = fechaPago.substring(0,4) + “-“fechaPago.substring(4,6) + “-“
fechaPago.substring(6);
                  oDatos.oCabFact.setFechaPago(fecha);
                  try{
                    oDatos.DML_ActualizarFactura();
                    info("Factura "+ numFactura + " grabada");
                  }catch(SQLException se){
                     info("Error al Actualizar...");
                  }
                  c=0;
               }
            }
      }
      return null;
   }
}
```

1.4.10 Clase manejadora de los datos de Canal.

Cada una de las tablas contienen clases que permiten manipular sus datos. Para insertar valores, se otorga a los atributos de la clase correspondiente, los valores que se ingresaron desde la interfaz, utilizando los métodos set.

Al realizar actualizaciones, previamente se debe extraer los datos de una fila, usando los métodos get.

```
package crm.vop;

public class Canal {
   private String cod_canal;
   private String desc_canal;
   private String tipo_canal;
   private float porcentaje_canal;

   public void setCodigo(String pCodCanal) {
      cod_canal=pCodCanal;
   }

   public String getCodigo() {
      return cod_canal;
   }

   public void setDescripcion(String pDescCanal) {
      desc_canal=pDescCanal;
   }

   public String getDescripcion() {
      return desc_canal;
   }

   public void setTipoCanal(String pTipoCanal) {
      tipo_canal=pTipoCanal;
   }

   public String getTipoCanal() {
      return tipo_canal;
   }

   public void setPorcentaje(float pPorcentaje) {
      porcentaje_canal=pPorcentaje;
   }

   public float getPorcentaje() {
      return porcentaje_canal;
   }
}
```

1.4.11 Código para imprimir reportes

El siguiente código permite imprimir reportes de comisiones con los datos mostrados en pantalla luego de realizar una consulta. Se muestra el cuadro de diálogo de la impresora y se envía cada uno de los datos con la sentencia drawString, que recibe la cadena a imprimir, la columna y la fila.

```
public String btnImprimir_action() {
        // TODO: Procesar la acción de clic de botón. El valor de devolución es un
                // nombre de caso de navegación en el que un valor nulo devolverá la misma
                página.

               Font fuente = new Font("Dialog", Font.PLAIN, 10);
        PrintJob pj;
        Graphics pagina;

        pj = Toolkit.getDefaultToolkit().getPrintJob(new Frame(), "SCAT", null);

        //recorrer el provider de vendedor
        boolean blHayRegistros=crm_vop_vendedorDataProvider.cursorFirst();
        String a=null;
        RowKey aRowKey=null;
        int cont = 0;
        String codVendedor; String nombreVendedor= null;

        while (blHayRegistros){
              a = new Integer(cont).toString();
              aRowKey=crm_vop_vendedorDataProvider.getRowKey(a);
              codVendedor = crm_vop_vendedorDataProvider.getValue("ven_codigo").toString();
              if (codVendedor.equals(cmbVendedor.getSelected().toString())){
                  nombreVendedor = crm_vop_vendedorDataProvider.getValue("emp_apellidos").toString();
              }
              blHayRegistros=crm_vop_vendedorDataProvider.cursorNext();
              cont++;
        }

         boolean blVendedor=false;
         if (chkVendedor.getValue() != null){
             blVendedor = Boolean.parseBoolean(chkVendedor.getValue().toString());
         }

         try{
            pagina = pj.getGraphics();
            pagina.setFont(fuente);
            pagina.setColor(Color.black);

            blHayRegistros=true;

            float valor; float valor_multa; float valor_com_canal; float valor_com_servicio; float valor_com_rotacion;
float valor_com_incentivo; String fecha;
            int fila = 130;
            cont = 1;
            String fechaMin=null; String fechaMax=null;
```

```
            if (txtFechaMin.getText() != null){
          fechaMin = txtFechaMin.getText().toString();
        }else{

        }
        if (txtFechaMax.getText() != null){
          fechaMax = txtFechaMax.getText().toString();
        }

        //Cabecera del reporte
        pagina.drawString("REPORTE DE COMISIONES", 200, 50);
        if (!blVendedor){
          pagina.drawString("Vendedor: " + nombreVendedor, 10, 90);
        }else{
          pagina.drawString("Vendedor: Todos", 10, 90);
        }
        if (txtFechaMin.getText() != null && txtFechaMax.getText() != null){
          pagina.drawString("Fecha desde: " + fechaMin + "hasta: " + fechaMax, 10, 100);
        }
        //Encabezado de la tabla
        pagina.drawString("Valor", 10, fila);
        pagina.drawString("Fecha", 70, fila);
        pagina.drawString("Valor multa", 130, fila);
        pagina.drawString("Valor com. canal", 210, fila);
        pagina.drawString("Valor com. servicio", 310, fila);
        pagina.drawString("Valor com. rotacion", 410, fila);
        pagina.drawString("Valor com. incentivo", 500, fila);
        blHayRegistros=crm_vop_comision_multaDataProvider.cursorFirst();

            while (blHayRegistros){
          a = new Integer(cont).toString();
          aRowKey=crm_vop_comision_multaDataProvider.getRowKey(a);
          valor =
Float.parseFloat(crm_vop_comision_multaDataProvider.getValue("CMS_VALOR_COMISIONES").toString());
          fecha = crm_vop_comision_multaDataProvider.getValue("CMS_FECHA_COMISIONES").toString();
          valor_multa =
Float.parseFloat(crm_vop_comision_multaDataProvider.getValue("CMS_VALOR_COMISIONES").toString());
          valor_com_canal =
Float.parseFloat(crm_vop_comision_multaDataProvider.getValue("CMS_VALOR_COMISIONES").toString());
          valor_com_servicio =
Float.parseFloat(crm_vop_comision_multaDataProvider.getValue("CMS_VALOR_COMISIONES").toString());
          valor_com_rotacion =
Float.parseFloat(crm_vop_comision_multaDataProvider.getValue("CMS_VALOR_COMISIONES").toString());
          valor_com_incentivo =
Float.parseFloat(crm_vop_comision_multaDataProvider.getValue("CMS_VALOR_COMISIONES").toString());
```

```
    fila = fila + 10;
                pagina.drawString(new Float(valor).toString(), 10, fila);
                pagina.drawString(fecha, 40, fila);
                pagina.drawString(new Float(valor_multa).toString(), 70, fila);
                pagina.drawString(new Float(valor_com_canal).toString(), 100, fila);
                pagina.drawString(new Float(valor_com_servicio).toString(), 130, fila);
                pagina.drawString(new Float(valor_com_rotacion).toString(), 160, fila);
                pagina.drawString(new Float(valor_com_incentivo).toString(), 190, fila);

                blHayRegistros=crm_vop_comision_multaDataProvider.cursorNext();
                cont++;
                info("Documento impreso correctamente");
            }

            pagina.dispose();
            pj.end();
        }catch(Exception e){
            info("La impresi\363n ha sido cancelada...");
        }

        return null;
    }
```

1.4.12 Clase Valida Numeros

Esta clase consta de métodos que permiten validar si un número es entero, flotante o es una cadena de caracteres que sólo contiene números.

```
public class ValidaNumeros {
    private static String numero = null;

    /** Creates a new instance of ValidaNumeros */
    public ValidaNumeros(String pNumero) {
        numero = pNumero;
    }

    public static boolean isNumeric(){
            try {
                Integer.parseInt(numero);
                    return true;
            } catch (NumberFormatException nfe){
                    return false;
            }
    }

    public static boolean isFloat(){
            try {
                    Float.parseFloat(numero);
                    return true;
            } catch (NumberFormatException nfe){
                    return false;
            }
    }


    public boolean isNumberString(){
        String validChars = "0123456789";
        boolean isNumber = true;

        for (int i = 0; i < numero.length() && isNumber; i++){
          char c = numero.charAt(i);
          if (validChars.indexOf(c) == -1){
            isNumber = false;
          }
          else{
            isNumber = true;
          }
        }
        return isNumber;
    }
}
```

1.5 Diccionario de datos

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM		**Fecha/diseño**		**Fecha/última actualización:**			
VERSION 1.0		Diciembre/2006		Febrero/2007			
Universidad de Guayaquil				**Autor:**			
Carrera de Ingeniería de Sistemas Computacionales				Grupo 2			
Nombre de Tabla:		**Base de Datos:**		**Tipo:**			
CRM_VOP_CONTRATO_CLIENTE		CRM		Transaccional			
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	CTC_CODCONTRATO	Number		Código de contrato de cliente	X		
2	CTC_CODCLIENTE	Number		Código del cliente	X	X	
3	CTC_FEC_INICIO	Date		Fecha inicio del contrato			
4	CTC_FEC_CADUCIDAD	Date		Fecha de caducidad del contrato			
5	CTC_FECHA_ MIN_CONTRATO	Date		Fecha mínima en que debe permanecer el cliente en cumplimiento con el contrato			
6	CTC_FECHA_RECISION	Date		Fecha en la que el cliente puede renunciar al contrato			
7	CTC_COD_VENDEDOR	Varchar2	6	Código del vendedor que atendió al cliente que firma el contrato		X	
8	CTC_COD_FORMA_PAGO	Varchar2	3	Código de la forma de pago que utilizará el cliente		X	
9	CTC_TIPO_CONTRATO	Varchar2	10	Define el tipo de contrato			
10	CTC_CODIGO_CICLO	Number		Fecha en que el cliente decide dejar de usar el servicio			X

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM		**Fecha/diseño**		**Fecha/última actualización:**			
VERSION 1.0		Diciembre/2006		Febrero/2007			
Universidad de Guayaquil					**Autor:**		
Carrera de Ingeniería de Sistemas Computacionales					Grupo 2		
Nombre de Tabla:		**Base de Datos:**		**Tipo:**			
CRM_VOP_SERV_CONT		CRM		Transaccional			
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	SEC_COD_CONTRATO	Number		Código de contrato de cliente	X	X	
2	SEC_CODCLIENTE	Number		Código del cliente	X	X	
3	SEC_CODIGO_SERV	Number		Código de servicio contratado	X	X	
4	SEC_FECHA_ACTIVACION	Date		Fecha de activación del servicio			
5	SEC_FECHA_FINAL	Date		Fecha final de disponibilidad del servicio			
6	SEC_PRECIO_SERV	Number		Específica el precio por servicio			
7	SEC_ESTADO	Varchar2	1	Específica el estado del servicio en el contrato			

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM		**Fecha/diseño**		**Fecha/última actualización:**			
VERSION 1.0		Diciembre/2006		Febrero/2007			
Universidad de Guayaquil					**Autor:**		
Carrera de Ingeniería de Sistemas Computacionales					Grupo 2		
Nombre de Tabla:		**Base de Datos:**		**Tipo:**			
CRM_VOP_VENTA		CRM		Transaccional			
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	VTS_CODIGO_VENTA	Number		Código de venta	X		
2	VTS_CODIDO_ CONTRATO	Number		Código del contrato		X	
3	VTS_CODIGO_CANAL	Varchar2	3	Código del canal		X	
4	VTS_CODIGO_ FORMA_PAGO	Varchar2	3	Código respectivo de la forma de pago		X	
5	VTS_CODIGO_AGENCIA	Number	4	Código de agencia		X	
6	VTS_VALOR	Number	6,2	Valor de la venta			
7	VTS_FECHA	Date		Fecha de realización de la venta			

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM	**Fecha/diseño**	**Fecha/última actualización:**					
VERSION 1.0	Diciembre/2006	Febrero/2007					
Universidad de Guayaquil				**Autor:**			
Carrera de Ingeniería de Sistemas Computacionales				Grupo 2			
Nombre de Tabla:	**Base de Datos:**	**Tipo:**					
CRM_VOP_VENT_VEND_SERV	CRM	Transaccional					
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	VVS_CODIGO_VENTA	Number		Código de venta	X	X	
2	VVS_SECUENCIA	Number		Secuencial por cada servicio registrado en una venta			
3	VVS_CODIGO_SERVICIO	Number		Código de servicio		X	

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM		**Fecha/diseño**		**Fecha/última actualización:**			
VERSION 1.0		Diciembre/2006		Febrero/2007			
Universidad de Guayaquil					**Autor:**		
Carrera de Ingeniería de Sistemas Computacionales					Grupo 2		
Nombre de Tabla:		**Base de Datos:**		**Tipo:**			
CRM_VOP_COMISION_MULTA		CRM		Transaccional			
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	CMS_COD_COMISIONES	Varchar2	8	Código de comisiones	X		
2	CMS_COD_VENDEDOR	Varchar2	10	Código de vendedor		X	
3	CMS_DESCRIPCION	Varchar2	100	Descripción: comisión o multa			
4	CMS_VALOR_COMISIONES	Number	6,2	Valor de la comisión			
5	CMS_FECHA_COMISIONES	Date		Fecha de comisión			
6	CMS_VALOR_MULTA	Number	6,2	Valor total de las multas			X
7	CMS_VALOR_COM_CANAL	Number	6,2	Valor total de comisión por canal			X
8	CMS_VALOR_ COM_SERVICIO	Number	6,2	Valor total de comisión por servicios costosos			x

9	CMS_VALOR_ COM_ROTACION	Number	6,2	Valor total de comisión por venta de servicios con rotación baja			X
10	CMS_VALOR_ COM_INCENTIVO	Number	6,2	Valor total de comisión por cumplimiento de metas			X

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM		**Fecha/diseño**		**Fecha/última actualización:**			
VERSION 1.0		Diciembre/2006		Febrero/2007			
Universidad de Guayaquil					**Autor:**		
Carrera de Ingeniería de Sistemas Computacionales							
Nombre de Tabla:		**Base de Datos:**		**Tipo:**			
CRM_ADM_PARAMETROS		CRM					
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	PRM_PORC_MULTA	Number	4,2	Porcentaje a descontar a los vendedores por recisión del contrato del cliente			
2	PRM_PORC_INCENTIVO_ANUAL	Number	4,2	Porcentaje a pagar a los vendedores por metas logradas			
3	PRM_DIA_COMISIONA	Number	2	Inicio del período establecido para pagar comisiones			
4	PRM_INTERES_MORA	Number	4,2	Porcentaje de interés por mora a cobrar a los clientes que adeudan.			
5	PRM_MES_ CAMBIO_CONTRATO	Number	2	Número de mes mínimo en que el cliente puede cambiar los servicios de su contrato.			
6	PRM_PORC_ CUOTA1_CREDITO	Number	4,2	Porcentaje mínimo de la primera cuota al establecer forma de pago a crédito			
7	PRM_MESES_ CADUCA_SERVICIO	Number	2	Especifica el número de meses en que va a caducar el servicio			

8	PRM_DIAS_CADUCA_FACTURA	Number	3	Especifica el número de meses en que la factura generada debe caducar			
9	PRM_VALOR_SERV_ MIN_VIP	Number	6,2	Valor mínimo del servicio para pagar comisión al vendedor que realice la venta			
10	PRM_PORC_ COMISION_SERVICIO	Number	4,2	Porcentaje a pagar a los vendedores que vendan un servicio con valor mayor al establecido para comisionar			
11	PRM_PORC_ COMISION_ROTACION	Number	4,2	Porcentaje a pagar a los vendedores que vendan servicios de menor rotación			

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM		**Fecha/diseño**		**Fecha/última actualización:**			
VERSION 1.0		Diciembre/2006		Febrero/2007			
Universidad de Guayaquil					**Autor:**		
Carrera de Ingeniería de Sistemas Computacionales					Grupo 2		
Nombre de Tabla:		**Base de Datos:**		**Tipo:**			
CRM_VOP_CAB_FACTURA		CRM		Transaccional			
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	CFAC_NUM_FACTURA	Number		Describe el número de la factura	X		
2	CFAC_COD_CLIENTE	Number		Describe el código del cliente		X	
3	CFAC_FECHA_EMISION	Date		Fecha emision de la factura			
4	CFAC_FECHA_VENCIMIENTO	Date		Fecha vencimiento de la factura			
5	CFAC_FECHA_PAGO	Date		Fecha en que canceló por completo la factura			
6	CFAC_VALOR_ABONO	Number	6,2	Valor que el cliente abona para pagar la factura			
7	CFAC_ESTADO	Varchar2	1	Determina el estado de la factura, si ésta ha sido cancelada (C), si hay algún abono (A) o si está pendiente (P) de pago			

8	CFAC_SUBTOTAL	Number	6,2	Subtotal de la Factura			
9	CFAC_IVA	Number	6,2	Monto IVA de la factura			
10	CFAC_ICE	Number	6,2	Monto ICE de la factura			
11	CFAC_RECARGO	Number	6,2	Valor de recargo en la factura producto del calculo del interés por mora			
12	CFAC_DESCUENTO	Number	6,2	Valor de descuento en la factura			
13	CFAC_TOTAL	Number	10,2	Valor total de la Factura			
14	CFAC_OBSERVACION	Varchar2	30	Glosa de observación referente a la factura emitida			

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM		**Fecha/diseño**		**Fecha/última actualización:**			
VERSION 1.0		Diciembre/2006		Febrero/2007			
Universidad de Guayaquil					**Autor:**		
Carrera de Ingeniería de Sistemas Computacionales					Grupo 2		
Nombre de Tabla:		**Base de Datos:**		**Tipo:**			
CRM_VOP_DET_FACTURA		CRM		Transaccional			
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	DFAC_NUM_FACTURA	Number		Describe el número de la factura que procede este detalle	X	X	
2	DFAC_NUM_ITEM	Varchar2	3	Valor secuencial por factura	X		
3	DFAC_ COD_SERVICIO	Number		Describe el código del Item o servicio contenido en la factura		X	
4	DFAC_CANTIDAD_SERVICIO	Number	3	Valor de Cantidad del servicio facturado			
5	DFAC_VALOR_SERVICIO	Number	6,2	Valor total del servicio producto del precio por la cantidad			

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM	**Fecha/diseño**	**Fecha/última actualización:**					
VERSION 1.0	Diciembre/2006	Febrero/2007					
Universidad de Guayaquil				**Autor:**			
Carrera de Ingeniería de Sistemas Computacionales				Grupo 2			
Nombre de Tabla:	**Base de Datos:**	**Tipo:**					
CRM_VOP_CAB_SOLICITUD	CRM	Transaccional					
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	CSL_NUM_SOLICITUD	Number		Describe el número de la solicitud	X		
2	CSL_COD_CLIENTE	Number		Describe el código del cliente		X	
3	CFAC_FECHA_EMISION	Date		Fecha emision de la solicitud			
4	CFAC_SUBTOTAL_ SOLICITUD	Number	6,2	Subtotal de la Solicitud			
5	CFAC_IVA	Number	6,2	Monto IVA de la Solicitud			
6	CFAC_ICE	Number	6,2	Monto ICE de la Solicitud			
7	CFAC_TOTAL_ SOLICITUD	Number	10,2	Valor total de la Solicitud			

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM		**Fecha/diseño**		**Fecha/última actualización:**			
VERSION 1.0		Diciembre/2006		Febrero/2007			
Universidad de Guayaquil					**Autor:**		
Carrera de Ingeniería de Sistemas Computacionales					Grupo 2		
Nombre de Tabla:		**Base de Datos:**		**Tipo:**			
CRM_VOP_DET_SOLICITUD		CRM		Transaccional			
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	DSL_NUM_SOLICITUD	Number		Describe el número de la solicitud que procede este detalle	X	X	
2	DSL_NUM_ITEM	Varchar2	3	Valor secuencial por factura	X		
3	DSL_ COD_SERVICIO	Number		Describe el código del Item o servicio contenido en la solicitud		X	
4	DSL_CANTIDAD_SERVICIO	Number	3	Valor de Cantidad del servicio solicitado			
5	DSL_VALOR_SERVICIO	Number	6,2	Valor total del servicio producto del precio por la cantidad			

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM		**Fecha/diseño**		**Fecha/última actualización:**			
VERSION 1.0		Diciembre/2006		Febrero/2007			
Universidad de Guayaquil					**Autor:**		
Carrera de Ingeniería de Sistemas Computacionales					Grupo 2		
Nombre de Tabla:		**Base de Datos:**		**Tipo:**			
CRM_VOP_CICLO_FACT		CRM		Referencial			
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	CIF_CODIGO	Number		Código del ciclo	X		
2	CIF_DIA_INICIO_CICLO	Number	2,0	Día inicio del ciclo			
3	CIF_DIA_FIN_CICLO	Number	2,0	Día fin del ciclo			
4	CIF_DESCRIPCION	Varchar2	40	Descripción del ciclo			
5	CIF_DIA_INICIO_FACT	Number	2,0	Día en que se empieza la facturación			
6	CIF_DIA_FIN_FACT	Number	2,0	Día final en que se deja de facturar			
7	CIF_NUM_DIAS_FACT	Number	2,0	Número de días a realizar facturación			

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM		**Fecha/diseño**		**Fecha/última actualización:**			
VERSION 1.0		Diciembre/2006		Febrero/2007			
Universidad de Guayaquil					**Autor:**		
Carrera de Ingeniería de Sistemas Computacionales					Grupo 2		
Nombre de Tabla:		**Base de Datos:**		**Tipo:**			
CRM_VOP_VENDEDOR		CRM		Referencial			
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	VEN_CODIGO	Varchar2	6	Código del Vendedor	X		
2	VEN_COD_ TIPO_VENDEDOR	Varchar2	2	Código del tipo de Vendedor		X	
3	VEN_COD_RANGO_VENTAS	Varchar2	4	Código que identifica el rango de ventas		X	
4	VEN_COD_CANAL	Varchar2	3	Código que identifica el canal		X	
5	VEN_COMISIONES	Number	6,2	Comisión a pagar al vendedor			
6	VEN_ESTADO	Varchar2	1	Identifica si el vendedor está A:Activo, I: Inactivo para realizar ventas			

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM	**Fecha/diseño**	**Fecha/última actualización:**					
VERSION 1.0	Diciembre/2006	Febrero/2007					
Universidad de Guayaquil				**Autor:**			
Carrera de Ingeniería de Sistemas Computacionales				Grupo 2			
Nombre de Tabla:	**Base de Datos:**	**Tipo:**					
CRM_VOP_TIPO_VENDEDOR	CRM	Referencial					
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	TIV_COD_TIPO_VENDEDOR	Varchar2	2	Código del tipo de vendedor	X		
2	TIV_DESCRIPCION_ TIPO_VENDEDOR	Varchar2	50	Identifica el detalle por cada tipo de vendedor			

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM	**Fecha/diseño**	**Fecha/última actualización:**					
VERSION 1.0	Diciembre/2006	Febrero/2007					
Universidad de Guayaquil				**Autor:**			
Carrera de Ingeniería de Sistemas Computacionales				Grupo 2			
Nombre de Tabla:	**Base de Datos:**	**Tipo:**					
CRM_VOP_ TELEFONO_VENDEDOR	CRM	Referencial					
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	TLV_CODIGO	Number	10	Código del Vendedor	X		
2	TLV_DESCRIPCION	Number	15	Código del tipo de Vendedor			
3	TLV_CODIGO_VENDEDOR	Varchar2	10	Código que identifica el rango de ventas		X	
4	TLV_CODIGO_TIPO_TLF	Varchar2	2	Código que identifica el canal		X	

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM		**Fecha/diseño**		**Fecha/última actualización:**			
VERSION 1.0		Diciembre/2006		Febrero/2007			
Universidad de Guayaquil					**Autor:**		
Carrera de Ingeniería de Sistemas Computacionales					Grupo 2		
Nombre de Tabla:		**Base de Datos:**		**Tipo:**			
CRM_VOP_CANAL		CRM		Referencial			
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	CNL_CODIGO	Varchar2	3	Código del canal	X		
2	CNL_DESCRIPCIONL	Varchar2	50	Descripción del canal			
3	CNL_TIPO_CANAL	Varchar2	1	Identifica el tipo de canal			
4	CNL_PORC_ COMISION_CANAL	Number	4,2	Define el porcentaje de comisión por canal			

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM		**Fecha/diseño**		**Fecha/última actualización:**			
VERSION 1.0		Diciembre/2006		Febrero/2007			
Universidad de Guayaquil					**Autor:**		
Carrera de Ingeniería de Sistemas Computacionales					Grupo 2		
Nombre de Tabla:		**Base de Datos:**		**Tipo:**			
CRM_VOP_RANGO_VENTAS		CRM		Referencial			
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	RNV_COD_RANGO_VENTAS	Varchar2	4	Código de rango ventas aplicado al pago de comisiones	X		
2	RNV_LIMITE_INFERIOR	Number	6,2	Identifica límite inferior correspondiente al rango de venta			
3	RNV_LIMITE_SUPERIOR	Number	6,2	Identifica límite superior correspondiente al rango de ventas			
4	RNV_DESCRIPCION	Varchar2	20	Específica información del rango de ventas			
5	RNV_PORCENTAJE_ COMISION	Number	2,2	Identifica el porcentaje de comisión cancelado al vendedor			
6	RNV_FECHA_ INICIAL_RANGO	Date		Identifica la fecha en que está asignado el rango de ventas			
7	RNV_FECHA_ FINAL_RANGO	Date		Identifica la fecha hasta la cual está en vigencia el rango de ventas asignado			
8	RNV_TIPO_META	Varchar2	1	Identifica metas mensuales, anuales			

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM		**Fecha/diseño**		**Fecha/última actualización:**			
VERSION 1.0		Diciembre/2006		Febrero/2007			
Universidad de Guayaquil					**Autor:**		
Carrera de Ingeniería de Sistemas Computacionales					Grupo 2		
Nombre de Tabla:		**Base de Datos:**		**Tipo:**			
CRM_VOP_OFERTA		CRM		Referencial			
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	OFR_COD_OFERTA	Varchar2	4	Identifica el código de la oferta disponible	X		
2	OFR_DESCRIPCION_ OFERTA	Varchar2	50	Identifica la descripción de la oferta			
3	OFR_MES_INICIAL	Number		Identifica la fecha desde que está vigente la oferta			
4	OFR_MES_FINAL	Number		Identifica la fecha hasta que estará vigente la oferta			
5	OFR_DESCUENTO	Number	12, 2	Identifica el descuento que se le va a otorgar al cliente según la oferta seleccionada			
6	OFR_TOTAL	Number	12, 2	Valor q identifica el total a cancelar después de realizado el descuento			

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM	**Fecha/diseño**	**Fecha/última actualización:**					
VERSION 1.0	Diciembre/2006	Marzo/2007					
Universidad de Guayaquil				**Autor:**			
Carrera de Ingeniería de Sistemas Computacionales				Grupo 2			
Nombre de Tabla:	**Base de Datos:**	**Tipo:**					
CRM_VOP_ OFERTA_SERVICIO	CRM	Referencial					
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	OSR_NUM_ITEM	Number		Identifica el número secuencial del item que forma parte del paquete promocional	X		
2	OSR_COD_OFERTA	Varchar2	4	Identifica el código de la oferta			
3	OSR_COD_ITEM	Number		Código del servicio que forma parte del paquete promocional			
4	OSR_PRECIO_ITEM	Number		Identifca el precio del item por servicio			

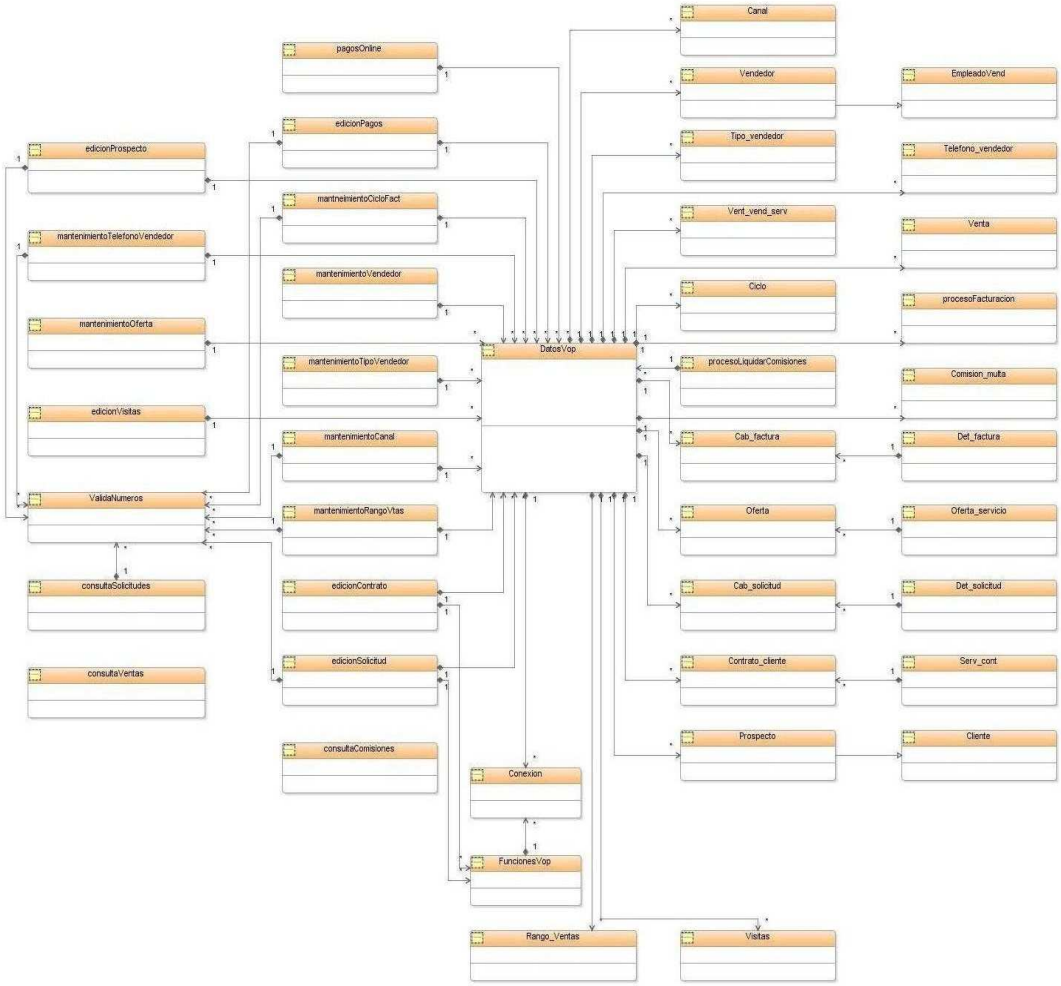
Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM		**Fecha/diseño**		**Fecha/última actualización:**			
VERSION 1.0		Diciembre/2006		Febrero/2007			
Universidad de Guayaquil					**Autor:**		
Carrera de Ingeniería de Sistemas Computacionales					Grupo 2		
Nombre de Tabla:		**Base de Datos:**		**Tipo:**			
CRM_VOP_VISITAS		CRM		Referencial			
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	VST_CONTROL	Number		Secuencial que identifica las visitas realizadas por determinado cliente y determinado servicio	X		
2	VST_ID_VISITA	Number		Código de la visita	X		
3	VST_CODIGO_CLIENTE	Number		Código del cliente	X	X	
4	VST_OBSERVACION	Varchar2	100	Específica el motivo de la visita o el porqué se encuentra pendiente			
5	VST_CODIGO_ VENDEDOR	Varchar2	10	Código del vendedor		X	
6	VST_CODIGO_CANAL	Varchar2	3	Código de canal		X	
7	VST_FECHA_ ASIGNACION	Date		Fecha de asignación para la visita			
8	VST_FECHA_VISITA	Date		Fecha en que se realiza la visita			
9	VST_ESTADO	Varchar2	1	Estado de la visita asignada. V: Visitado, P: Pendiente			

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM	**Fecha/diseño**		**Fecha/última actualización:**				
VERSION 1.0	Diciembre/2006		Marzo/2007				
Universidad de Guayaquil				**Autor:**			
Carrera de Ingeniería de Sistemas Computacionales				Grupo 2			
Nombre de Tabla:	**Base de Datos:**		**Tipo:**				
CRM_VOP_PROSPECTO	CRM		Referencial				
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	PRO_CODIGO_ PROSPECTO	Number		Código del prospecto	X		
2	PRO_ TIPO_PROSPECTO	Number		Hace referencia al tipo de cliente		X	
3	PRO_NOMBRES	Varchar2	60	Identifica los nombres del prospecto			
4	PRO_APELLIDOS	Varchar2	60	Identifica los apellidos del prospecto			
5	PRO_CEDULA	Varchar2	10	Identifica el número de cédula del prospecto			
6	PRO_PASAPORTE	Varchar2	30	Identifica el número de pasaporte del prospecto			
7	PRO_SEXO	Varchar2	1	Identifica el género al que pertenece el prospecto			
8	PRO_ESTADO_CIVIL	Varchar2	2	Identifica el estado civil del prospecto			
9	PRO_FEC_NAC	Date		Identifica fecha de nacimiento del prospecto			

10	PRO_ACTIVIDAD	Number		Identifica la actividad mediante el al que se dedica el prospecto		X	
11	PRO_CODIGO_NAC	Varchar2	3	Identifica el código de nacionalidad del prospecto		X	
12	PRO_ESTADO	Varchar2	1	Identifica el estado del prospecto. A: Activo, I: Inactivo			
13	PRO_TIPO_ CLIENTE_JURIDICO	Number		Identifica el tipo de cliente jurídico del prospecto			
14	PRO_REP_LEG	Varchar2	60	Identifica al representante legal de la organización			

Módulo de Ventas Operacionales							
SISTEMA INTEGRA				Página 1/1			
CRM	**Fecha/diseño**	**Fecha/última actualización:**					
VERSION 1.0	Diciembre/2006	Marzo/2007					
Universidad de Guayaquil				**Autor:**			
Carrera de Ingeniería de Sistemas Computacionales				Grupo 2			
Nombre de Tabla:	**Base de Datos:**	**Tipo:**					
CRM_VOP_ ATENCION_LLAMADAS	CRM	Referencial					
Descripción							
Descripción de Campos							
No	Nombre	Tipo	Long.	Descripción	PK	FK	Nulo
1	ATL_COD_LLAMADA	Varchar2	4	Código de la llamada	X		
2	ATL_CI	Varchar2	10	Identifica el número de cédula del cliente que realizada llamada			
3	ATL_NOMBRES	Varchar2	50	Identifica los nombres del cliente que realiza la llamada			
4	ATL_APELLIDOS	Varchar2	50	Identifica los apellidos del cliente que realiza la llamada			
5	ATL_OBSERVACION	Varchar2	50	Identifica el motivo de la llamada			
6	ATL_COD_SERVICIO	Number		Identifica el código del servicio que posee el cliente		X	
7	ATL_PRIORIDAD	Number	2	Identifica la prioridad de ingreso de llamada por servicio que tiene el cliente			
8	ATL_NUM_IVR	Number	2	Identifica el número de marcación IVR			
9	ATL_ESTADO	Varchar2	1	Identifica el estado de la llamada			

1.6 Diagrama de clases



Este diagrama de clases simplificado, indica las relaciones entre clases correspondientes al módulo de ventas operacionales, sin mostrar los atributos y métodos.

Para ampliar la información: Ver Anexo A.

1.7. Diagrama entidad-relación

El diagrama entidad-relación correspondiente al módulo de ventas operacionales, como es mostrado en la figura del Anexo B., contiene las tablas relacionadas con las operaciones de ventas de la empresa, información de contratos, comisiones.

Para implementar este módulo es necesaria la información de otros módulos tales como: Marketing Operacional y Servicio al Cliente Operacional, los mismos que proporcionan información relativa a servicios y clientes, respectivamente; y con el módulo de Administración e Integración del CRM, que proporcionan las tablas de datos parametrizados para todo el sistema.

Ver Anexo B.

1.8 PL/SQL

1.8.1.Funciones almacenadas

1.8.1.1.Función VOP_F_INSERTAR_CAB_SOLICITUD

La siguiente función inserta los datos de cabecera de solicitud retornando el código para utilizarlo en la inserción de los datos del detalle de dicha solicitud.

```
CREATE OR REPLACE FUNCTION VOP_F_INSERTAR_CAB_SOLICITUD(
  VI_COD_CLIENTE  IN CRM_VOP_CAB_SOLICITUD.CSL_COD_CLIENTE%TYPE,
  VI_FECHA_EMISION IN CRM_VOP_CAB_SOLICITUD.CSL_FECHA_EMISION%TYPE,
  VI_SUBTOTAL_SOLICITUD IN
CRM_VOP_CAB_SOLICITUD.CSL_SUBTOTAL_SOLICITUD%TYPE,
  VI_IVA   IN CRM_VOP_CAB_SOLICITUD.CSL_IVA%TYPE,
  VI_ICE   IN CRM_VOP_CAB_SOLICITUD.CSL_ICE%TYPE,
  VI_TOTAL_SOLICITUD IN CRM_VOP_CAB_SOLICITUD.CSL_TOTAL_SOLICITUD%TYPE,
  VO_NUM_SOLICITUD OUT CRM_VOP_CAB_SOLICITUD.CSL_NUM_SOLICITUD%TYPE)
RETURN CRM_VOP_CAB_SOLICITUD.CSL_NUM_SOLICITUD%TYPE
IS
  VL_NUM_SOLICITUD CRM_VOP_CAB_SOLICITUD.CSL_NUM_SOLICITUD%TYPE;
BEGIN

  SELECT MAX(CSL_NUM_SOLICITUD) INTO VL_NUM_SOLICITUD
  FROM CRM_VOP_CAB_SOLICITUD;

  IF VL_NUM_SOLICITUD IS NULL THEN
    VL_NUM_SOLICITUD:=0;
  END IF;

  VL_NUM_SOLICITUD:=VL_NUM_SOLICITUD+1;
```

```
 INSERT INTO CRM_VOP_CAB_SOLICITUD
 (
  CSL_NUM_SOLICITUD,
  CSL_COD_CLIENTE,
  CSL_FECHA_EMISION,
  CSL_SUBTOTAL_SOLICITUD,
  CSL_IVA,
  CSL_ICE,
  CSL_TOTAL_SOLICITUD
 )
 VALUES (
  VL_NUM_SOLICITUD,
  VI_COD_CLIENTE,
  VI_FECHA_EMISION,
  VI_SUBTOTAL_SOLICITUD,
  VI_IVA,
  VI_ICE,
  VI_TOTAL_SOLICITUD
 );
 VO_NUM_SOLICITUD:=VL_NUM_SOLICITUD;
 RETURN VO_NUM_SOLICITUD;
END VOP_F_INSERTAR_CAB_SOLICITUD;
```

1.8.1.2. Función VOP_F_INSERTAR_OFERTA

La siguiente función inserta los datos de ofertas retornando el código para utilizarlo en la inserción de los datos de los servicios que contiene dicha oferta.

```
CREATE OR REPLACE FUNCTION VOP_F_INSERTAR_OFERTA(
    VI_DESCRIPCION_OFERTA IN
CRM_VOP_OFERTA.OFR_DESCRIPCION_OFERTA%TYPE,
    VI_MES_INICIAL       IN  CRM_VOP_OFERTA.OFR_MES_INICIAL%TYPE,
    VI_MES_FINAL         IN  CRM_VOP_OFERTA.OFR_MES_FINAL%TYPE,
    VI_DESCUENTO          IN  CRM_VOP_OFERTA.OFR_DESCUENTO%TYPE,
    VI_TOTAL            IN  CRM_VOP_OFERTA.OFR_TOTAL%TYPE,
    VO_COD_OFERTA        OUT CRM_VOP_OFERTA.OFR_COD_OFERTA%TYPE)

RETURN CRM_VOP_OFERTA.OFR_COD_OFERTA%TYPE
IS
  VL_COD_OFERTA   CRM_VOP_OFERTA.OFR_COD_OFERTA%TYPE;
BEGIN

  SELECT MAX(OFR_COD_OFERTA) INTO VL_COD_OFERTA
  FROM CRM_VOP_OFERTA;

  IF VL_COD_OFERTA IS NULL THEN
    VL_COD_OFERTA:=0;
  END IF;

  VL_COD_OFERTA:=VL_COD_OFERTA+1;

  INSERT INTO CRM_VOP_OFERTA
  (
    OFR_COD_OFERTA,
    OFR_DESCRIPCION_OFERTA,
    OFR_MES_INICIAL,
    OFR_MES_FINAL,
    OFR_DESCUENTO,
    OFR_TOTAL
  )
  VALUES (
    VL_COD_OFERTA,
    VI_DESCRIPCION_OFERTA,
    VI_MES_INICIAL,
    VI_MES_FINAL,
    VI_DESCUENTO,
    VI_TOTAL
  );

  VO_COD_OFERTA:=VL_COD_OFERTA;
  RETURN VO_COD_OFERTA;

END VOP_F_INSERTAR_OFERTA;
```

1.8.1.3. Función VOP_F_VERIFICAR_CEDULA_RUC

La siguiente función verifica cédula y ruc del cliente dependiendo del tipo de cliente sea persona natural o jurídica.

Para persona natural valida la cédula o ruc y para la persona jurídica valida el ruc correspondiente y retorna CORRECTO: cuando el número de cédula/ruc sean válidos o INCORRECTO: cuando el número de cédula/ruc sea inválido.

```
CREATE OR REPLACE FUNCTION VOP_F_VERIFICAR_CEDULA_RUC(
  VI_CEDULA_RUC IN  VARCHAR2,
  VI_TIPO_PERSONA IN CRM_VOP_PROSPECTO.PRO_TIPO_PROSPECTO%TYPE,
  VO_RESULTADO  OUT VARCHAR2
  )RETURN VARCHAR2 IS
  VL_NUM NUMBER;
  RESULT1 NUMBER;
  RESULT2 NUMBER;
  I NUMBER;
  ACUM NUMBER;
  AUX NUMBER;
  VL_CEDULA_RUC VARCHAR2(16);
  VL_TIPO_PERSONA VARCHAR2(1);
BEGIN

  RESULT1 := 0;
  RESULT2 := 0;
  ACUM := 0;

  VL_CEDULA_RUC := TRIM(VI_CEDULA_RUC);

  FOR I IN 0..8 LOOP
     VL_NUM := TO_NUMBER(SUBSTR(VI_CEDULA_RUC,I+1,1));
     IF ((I MOD 2) = 1) THEN
        RESULT2 := RESULT2 + VL_NUM;
     ELSE
        AUX := VL_NUM*2;
        IF (VL_NUM*2 >= 10) THEN
           AUX := AUX - 9;
        END IF;
        RESULT1 := RESULT1 + AUX;
     END IF;
  END LOOP;

  WHILE ((RESULT1 + RESULT2) > ACUM) LOOP
     ACUM := ACUM + 10;
  END LOOP;
```

```
  IF VI_TIPO_PERSONA = 1 THEN
     VL_TIPO_PERSONA:='N';
  ELSE
     VL_TIPO_PERSONA:='J';
  END IF;

IF VL_TIPO_PERSONA='N' THEN
    IF (LENGTH(VL_CEDULA_RUC) = 10) THEN
       IF ((ACUM-RESULT1-RESULT2) = TO_NUMBER(SUBSTR(VI_CEDULA_RUC,10,1)))
THEN
          VO_RESULTADO := 'CORRECTO';
       ELSE
          VO_RESULTADO := 'INCORRECTO';
      END IF;
    ELSE
       VO_RESULTADO := 'INCORRECTO';
    END IF;
  END IF;

  IF (VL_TIPO_PERSONA='N'  AND VO_RESULTADO='INCORRECTO') OR
VL_TIPO_PERSONA='J' THEN
    IF (LENGTH(VL_CEDULA_RUC) = 13) THEN
       IF ((ACUM-RESULT1-RESULT2) = TO_NUMBER(SUBSTR(VI_CEDULA_RUC,10,1))
AND SUBSTR(VI_CEDULA_RUC,11,3) = '001') THEN
          VO_RESULTADO := 'CORRECTO';
       ELSE
          VO_RESULTADO := 'INCORRECTO';
       END IF;
    ELSE
       VO_RESULTADO := 'INCORRECTO';
    END IF;
  END IF;

  RETURN(VO_RESULTADO);

END VOP_F_VERIFICAR_CEDULA_RUC;
/
```

1.8.1.4. Función VOP_F_VERIFICAR_FECHA_NAC

La siguiente función verifica la fecha de nacimiento devolviendo CORRECTO: si la persona es mayor a 18 años o INCORRECTO: si es menor.

```
CREATE OR REPLACE FUNCTION VOP_F_VERIFICAR_FECHA_NAC(
  VI_FECHA      IN  DATE,
  VO_RESULTADO  OUT VARCHAR2
  )RETURN VARCHAR2 IS

BEGIN
  IF (MONTHS_BETWEEN(SYSDATE, VI_FECHA) >= 216) THEN
     VO_RESULTADO := 'CORRECTO';
  ELSE
     VO_RESULTADO := 'INCORRECTO';
  END IF;

  RETURN(VO_RESULTADO);
END VOP_F_VERIFICAR_FECHA_NAC;
/
```

1.8.2. Procedimientos almacenados

1.8.2.1 Procedimiento VOP_P_ACTUALIZAR_CANAL

El siguiente procedimiento actualiza los datos del canal seleccionado correspondiente a la tabla CRM_VOP_CANAL.

```
CREATE OR REPLACE PROCEDURE VOP_P_ACTUALIZAR_CANAL(

    VI_CODIGO       IN CRM_VOP_CANAL.CNL_CODIGO%TYPE,
    VI_DESCRIPCION   IN CRM_VOP_CANAL.CNL_DESCRIPCIONL%TYPE,
    VI_TIPO_CANAL    IN CRM_VOP_CANAL.CNL_TIPO_CANAL%TYPE,
    VI_PORC_COMISION IN CRM_VOP_CANAL.CNL_PORC_COMISION_CANAL%TYPE)

IS
BEGIN

 UPDATE CRM_VOP_CANAL SET
       CNL_DESCRIPCIONL = VI_DESCRIPCION,
       CNL_TIPO_CANAL = VI_TIPO_CANAL,
       CNL_PORC_COMISION_CANAL = VI_PORC_COMISION

 WHERE  CNL_CODIGO = VI_CODIGO;

END VOP_P_ACTUALIZAR_CANAL;
/
```

1.8.2.2. Procedimiento VOP_P_ACTUALIZAR_CICLO

El siguiente procedimiento actualiza los datos del ciclo seleccionado correspondiente a la tabla CRM_VOP_CICLO_FACT.

```
CREATE OR REPLACE PROCEDURE VOP_P_ACTUALIZAR_CICLO(
    VI_CODIGO              IN CRM_VOP_CICLO_FACT.CIF_CODIGO%TYPE,
    VI_DESCRIPCION         IN CRM_VOP_CICLO_FACT.CIF_DESCRIPCION%TYPE,
    VI_DIA_INICIO_CICLO IN CRM_VOP_CICLO_FACT.CIF_DIA_INICIO_CICLO%TYPE,
    VI_DIA_FIN_CICLO       IN CRM_VOP_CICLO_FACT.CIF_DIA_FIN_CICLO%TYPE,
    VI_DIA_INICIO_FACT     IN CRM_VOP_CICLO_FACT.CIF_DIA_INICIO_FACT%TYPE,
    VI_DIA_FIN_FACT        IN CRM_VOP_CICLO_FACT.CIF_DIA_FIN_FACT%TYPE,
    VI_NUM_DIAS_FACT       IN CRM_VOP_CICLO_FACT.CIF_NUM_DIAS_FACT%TYPE)
IS
BEGIN

  UPDATE  CRM_VOP_CICLO_FACT
   SET
    CIF_DESCRIPCION = VI_DESCRIPCION,
    CIF_DIA_INICIO_CICLO = VI_DIA_INICIO_CICLO,
    CIF_DIA_FIN_CICLO = VI_DIA_FIN_CICLO,
    CIF_DIA_INICIO_FACT = VI_DIA_INICIO_FACT,
    CIF_DIA_FIN_FACT = VI_DIA_FIN_FACT,
    CIF_NUM_DIAS_FACT = VI_NUM_DIAS_FACT

  WHERE CIF_CODIGO  = VI_CODIGO;

END VOP_P_ACTUALIZAR_CICLO;
/
```

1.8.2.3. Procedimiento VOP_P_ACTUALIZAR_OFERTA

El siguiente procedimiento actualiza los datos de la oferta seleccionada correspondiente a la tabla CRM_VOP_OFERTA.

```
CREATE OR REPLACE PROCEDURE VOP_P_ACTUALIZAR_OFERTA(

    VI_COD_OFERTA  IN CRM_VOP_OFERTA.OFR_COD_OFERTA%TYPE,
    VI_DESCRIPCION_OFERTA IN
CRM_VOP_OFERTA.OFR_DESCRIPCION_OFERTA%TYPE,
    VI_FECHA_INICIAL IN CRM_VOP_OFERTA.OFR_FECHA_INICIAL%TYPE,
    VI_FECHA_FINAL   IN CRM_VOP_OFERTA.OFR_FECHA_FINAL%TYPE,
    VI_COD_SERVICIO  IN CRM_VOP_OFERTA.OFR_COD_SERVICIO%TYPE)

IS
BEGIN

 UPDATE CRM_VOP_OFERTA SET
   OFR_COD_OFERTA = VI_COD_OFERTA,
   OFR_DESCRIPCION_OFERTA = VI_DESCRIPCION_OFERTA,
   OFR_FECHA_INICIAL = VI_FECHA_INICIAL,
   OFR_FECHA_FINAL = VI_FECHA_FINAL,
   OFR_COD_SERVICIO = VI_COD_SERVICIO

 WHERE OFR_COD_OFERTA = VI_COD_OFERTA;


END VOP_P_ACTUALIZAR_OFERTA;
/
```

1.8.2.4. Procedimiento VOP_P_ACTUALIZAR_PROSPECTO

El siguiente procedimiento actualiza los datos del prospecto seleccionado correspondiente a la tabla CRM_VOP_PROSPECTO.

```
CREATE OR REPLACE PROCEDURE VOP_P_ACTUALIZAR_PROSPECTO(
  VI_CODIGO_PROSPECTO IN
CRM_VOP_PROSPECTO.PRO_CODIGO_PROSPECTO%TYPE,
  VI_TIPO_PROSPECTO   IN
CRM_VOP_PROSPECTO.PRO_TIPO_PROSPECTO%TYPE,
  VI_NOMBRES          IN
CRM_VOP_PROSPECTO.PRO_NOMBRES%TYPE,
  VI_APELLIDOS        IN
CRM_VOP_PROSPECTO.PRO_APELLIDOS%TYPE,
  VI_CEDULA           IN CRM_VOP_PROSPECTO.PRO_CEDULA%TYPE,
  VI_PASAPORTE        IN
CRM_VOP_PROSPECTO.PRO_PASAPORTE%TYPE,
  VI_SEXO             IN CRM_VOP_PROSPECTO.PRO_SEXO%TYPE,
  VI_ESTADO_CIVIL     IN
CRM_VOP_PROSPECTO.PRO_ESTADO_CIVIL%TYPE,
  VI_FECHA_NAC        IN
CRM_VOP_PROSPECTO.PRO_FEC_NAC%TYPE,
  VI_ACTIVIDAD        IN
CRM_VOP_PROSPECTO.PRO_ACTIVIDAD%TYPE,
  VI_NACIONALIDAD     IN
CRM_VOP_PROSPECTO.PRO_CODIGO_NAC%TYPE,
  VI_ESTADO           IN CRM_VOP_PROSPECTO.PRO_ESTADO%TYPE,
  VI_TIPO_CLIENTE_JURIDICO IN
CRM_VOP_PROSPECTO.PRO_TIPO_CLIENTE_JURIDICO%TYPE,
  VI_REP_LEGAL        IN

CRM_VOP_PROSPECTO.PRO_REP_LEG%TYPE)
IS
BEGIN
 UPDATE CRM_VOP_PROSPECTO
  SET
    PRO_NOMBRES = VI_NOMBRES,
    PRO_APELLIDOS = VI_APELLIDOS,
    PRO_CEDULA = VI_CEDULA,
    PRO_PASAPORTE = VI_PASAPORTE,
    PRO_SEXO = VI_SEXO,
    PRO_ESTADO_CIVIL = VI_ESTADO_CIVIL,
    PRO_FEC_NAC = VI_FECHA_NAC,
    PRO_ACTIVIDAD = VI_ACTIVIDAD,
    PRO_CODIGO_NAC = VI_NACIONALIDAD,
    PRO_ESTADO = VI_ESTADO,
    PRO_TIPO_PROSPECTO = VI_TIPO_PROSPECTO,
    PRO_TIPO_CLIENTE_JURIDICO = VI_TIPO_CLIENTE_JURIDICO,
    PRO_REP_LEG = VI_REP_LEGAL
  WHERE PRO_CODIGO_PROSPECTO= VI_CODIGO_PROSPECTO;
END VOP_P_ACTUALIZAR_PROSPECTO;
/
```

1.8.2.5. Procedimiento VOP_P_ACTUALIZAR_RANGO_VENTAS

El siguiente procedimiento actualiza los datos del rango de ventas seleccionado correspondiente a la tabla CRM_VOP_RANGO_VENTAS.

```
CREATE OR REPLACE PROCEDURE VOP_P_ACTUALIZAR_RANGO_VENTAS(

    VI_COD_RANGO_VENTAS     IN
CRM_VOP_RANGO_VENTAS.RNV_COD_RANGO_VENTAS%TYPE,
    VI_LIMITE_INFERIOR      IN
CRM_VOP_RANGO_VENTAS.RNV_LIMITE_INFERIOR%TYPE,
    VI_LIMITE_SUPERIOR      IN
CRM_VOP_RANGO_VENTAS.RNV_LIMITE_SUPERIOR%TYPE,
    VI_PORCENTAJE_COMISION  IN
CRM_VOP_RANGO_VENTAS.RNV_PORCENTAJE_COMISION%TYPE,
    VI_FECHA_INICIAL_RANGO  IN
CRM_VOP_RANGO_VENTAS.RNV_FECHA_INICIAL_RANGO%TYPE,
    VI_FECHA_FINAL_RANGO    IN
CRM_VOP_RANGO_VENTAS.RNV_FECHA_FINAL_RANGO%TYPE,
    VI_TIPO_META            IN CRM_VOP_RANGO_VENTAS.RNV_TIPO_META%TYPE,
    VI_DESCRIPCION          IN CRM_VOP_RANGO_VENTAS.RNV_DESCRIPCION%TYPE)
IS
BEGIN

 UPDATE CRM_VOP_RANGO_VENTAS SET
   RNV_LIMITE_INFERIOR = VI_LIMITE_INFERIOR,
   RNV_LIMITE_SUPERIOR = VI_LIMITE_SUPERIOR,
   RNV_PORCENTAJE_COMISION = VI_PORCENTAJE_COMISION,
   RNV_FECHA_INICIAL_RANGO = VI_FECHA_INICIAL_RANGO,
   RNV_FECHA_FINAL_RANGO = VI_FECHA_FINAL_RANGO,
   RNV_TIPO_META = VI_TIPO_META,
   RNV_DESCRIPCION = VI_DESCRIPCION
 WHERE RNV_COD_RANGO_VENTAS = VI_COD_RANGO_VENTAS;

END VOP_P_ACTUALIZAR_RANGO_VENTAS;
/
```

1.8.2.6. Procedimiento VOP_P_ACTUALIZAR_TELF_VEND

El siguiente procedimiento actualiza los datos del teléfono del vendedor seleccionado correspondiente a la tabla CRM_VOP_TELEFONO_VENDEDOR.

```
CREATE OR REPLACE PROCEDURE VOP_P_ACTUALIZAR_TELF_VEND(

    VI_CODIGO               IN CRM_VOP_TELEFONO_VENDEDOR.TLV_CODIGO%TYPE,
    VI_DESCRIPCION      IN
CRM_VOP_TELEFONO_VENDEDOR.TLV_DESCRIPCION%TYPE,
    VI_CODIGO_VENDEDOR      IN
CRM_VOP_TELEFONO_VENDEDOR.TLV_CODIGO_VENDEDOR%TYPE,
    VI_CODIGO_TIPO_TLF      IN
CRM_VOP_TELEFONO_VENDEDOR.TLV_CODIGO_TIPO_TLF%TYPE)

IS
BEGIN

  UPDATE  CRM_VOP_TELEFONO_VENDEDOR SET
    TLV_CODIGO = VI_CODIGO,
    TLV_DESCRIPCION = VI_DESCRIPCION,
    TLV_CODIGO_VENDEDOR = VI_CODIGO_VENDEDOR,
    TLV_CODIGO_TIPO_TLF = VI_CODIGO_TIPO_TLF

        WHERE TLV_CODIGO = VI_CODIGO;


END VOP_P_ACTUALIZAR_TELF_VEND;
/
```

1.8.2.7. Procedimiento VOP_P_ACTUALIZAR_TIPO_VENDEDOR

El siguiente procedimiento actualiza los datos del tipo de vendedor seleccionado correspondiente a la tabla CRM_VOP_TIPO_VENDEDOR.

```
CREATE OR REPLACE PROCEDURE VOP_P_ACTUALIZAR_TIPO_VENDEDOR(

    VI_COD_TIPO_VENDEDOR         IN
CRM_VOP_TIPO_VENDEDOR.TIV_COD_TIPO_VENDEDOR%TYPE,
    VI_DESCRIPCION_TIPO_VENDEDOR  IN
CRM_VOP_TIPO_VENDEDOR.TIV_DESCRIPCION_TIPO_VENDEDOR%TYPE)

IS
BEGIN

 UPDATE CRM_VOP_TIPO_VENDEDOR SET
   TIV_COD_TIPO_VENDEDOR = VI_COD_TIPO_VENDEDOR,
   TIV_DESCRIPCION_TIPO_VENDEDOR = VI_DESCRIPCION_TIPO_VENDEDOR

       WHERE TIV_COD_TIPO_VENDEDOR = VI_COD_TIPO_VENDEDOR;


END VOP_P_ACTUALIZAR_TIPO_VENDEDOR;
/
```

1.8.2.8. Procedimiento VOP_P_ACTUALIZAR_VENDEDOR

El siguiente procedimiento actualiza los datos del vendedor seleccionado correspondiente a la tabla CRM_VOP_VENDEDOR.

```
CREATE OR REPLACE PROCEDURE VOP_P_ACTUALIZAR_VENDEDOR(

    VI_CODIGO   IN CRM_VOP_VENDEDOR.VEN_CODIGO%TYPE,
    VI_COD_TIPO_VENDEDOR  IN
CRM_VOP_VENDEDOR.VEN_COD_TIPO_VENDEDOR%TYPE,
    VI_COD_RANGO_VENTAS        IN
CRM_VOP_VENDEDOR.VEN_COD_RANGO_VENTAS%TYPE,
    VI_COD_CANAL   IN CRM_VOP_VENDEDOR.VEN_COD_CANAL%TYPE,
    VI_COMISIONES   IN CRM_VOP_VENDEDOR.VEN_COMISIONES%TYPE,
    VI_ESTADO       IN CRM_VOP_VENDEDOR.VEN_ESTADO%TYPE)


IS
BEGIN

 UPDATE CRM_VOP_VENDEDOR SET
  VEN_CODIGO = VI_CODIGO,
  VEN_COD_TIPO_VENDEDOR = VI_COD_TIPO_VENDEDOR,
  VEN_COD_RANGO_VENTAS = VI_COD_RANGO_VENTAS,
  VEN_COD_CANAL = VI_COD_CANAL,
  VEN_COMISIONES = VI_COMISIONES,
  VEN_ESTADO = VI_ESTADO

 WHERE VEN_CODIGO = VI_CODIGO;


END VOP_P_ACTUALIZAR_VENDEDOR;
/
```

1.8.2.9. Procedimiento VOP_P_ACTUALIZAR_OFERTA_SERV

El siguiente procedimiento permite registrar los servicios que pertenecen a una oferta, utilizando el código que tiene en la tabla CRM_VOP_OFERTA, y generando códigos para los números de item.

```
CREATE OR REPLACE PROCEDURE VOP_P_ACTUALIZAR_OFERTA_SERV(
    VI_NUM_ITEM      IN CRM_VOP_OFERTA_SERVICIO.OSR_NUM_ITEM%TYPE,
    VI_COD_OFERTA    IN CRM_VOP_OFERTA_SERVICIO.OSR_COD_OFERTA%TYPE,
    VI_COD_ITEM      IN CRM_VOP_OFERTA_SERVICIO.OSR_COD_ITEM%TYPE,
    VI_PRECIO_ITEM   IN CRM_VOP_OFERTA_SERVICIO.OSR_PRECIO_ITEM%TYPE)
IS
BEGIN
    UPDATE CRM_VOP_OFERTA_SERVICIO
    SET
      OSR_COD_OFERTA = VI_COD_OFERTA,
      OSR_COD_ITEM = VI_COD_ITEM,
      OSR_PRECIO_ITEM = VI_PRECIO_ITEM
    WHERE
      OSR_NUM_ITEM = VI_NUM_ITEM;
      --OSR_COD_OFERTA = VI_COD_OFERTA;
END VOP_P_ACTUALIZAR_OFERTA_SERV;
/
```

1.8.2.10. Procedimiento VOP_P_ACTUALIZAR_VISITA

El siguiente procedimiento actualiza los datos de la visita seleccionada correspondiente a la tabla CRM_VOP_VISITAS.

```
CREATE OR REPLACE PROCEDURE VOP_P_ACTUALIZAR_VISITA(
    VI_CONTROL          IN CRM_VOP_VISITAS.VST_CONTROL%TYPE,
    VI_ID_VISITA        IN CRM_VOP_VISITAS.VST_ID_VISITA%TYPE,
    VI_CODIGO_CLIENTE   IN CRM_VOP_VISITAS.VST_CODIGO_CLIENTE%TYPE,
    VI_FECHA_VISITA     IN CRM_VOP_VISITAS.VST_FECHA_VISITA%TYPE,
    VI_ESTADO           IN CRM_VOP_VISITAS.VST_ESTADO%TYPE,
    VI_OBSERVACION      IN CRM_VOP_VISITAS.VST_OBSERVACION%TYPE)
IS
BEGIN

 UPDATE CRM_VOP_VISITAS
  SET   VST_FECHA_VISITA   = VI_FECHA_VISITA,
        VST_ESTADO         = VI_ESTADO,
        VST_OBSERVACION    = VI_OBSERVACION
  WHERE VI_CONTROL         = VST_CONTROL
  AND   VI_ID_VISITA       = VST_ID_VISITA
  AND   VI_CODIGO_CLIENTE  = VST_CODIGO_CLIENTE;
END VOP_P_ACTUALIZAR_VISITA;
/
```

1.8.2.11. Procedimiento VOP_P_ELIMINAR_CANAL

El siguiente procedimiento elimina los datos del canal seleccionado correspondiente a la tabla CRM_VOP_CANAL.

```
CREATE OR REPLACE PROCEDURE VOP_P_ELIMINAR_CANAL (
                    VI_CODIGO IN CRM_VOP_CANAL.CNL_CODIGO%TYPE)
IS
BEGIN

  DELETE FROM CRM_VOP_CANAL
  WHERE CNL_CODIGO  = VI_CODIGO;

END VOP_P_ELIMINAR_CANAL;
/
```

1.8.2.12. Procedimiento VOP_P_ELIMINAR_CICLO

El siguiente procedimiento elimina los datos del ciclo de facturación seleccionado correspondiente a la tabla CRM_VOP_CICLO_FACT.

```
CREATE OR REPLACE PROCEDURE VOP_P_ELIMINAR_CICLO (
                    VI_CODIGO IN CRM_VOP_CICLO_FACT.CIF_CODIGO%TYPE)
IS
BEGIN

  DELETE FROM CRM_VOP_CICLO_FACT
  WHERE CIF_CODIGO  = VI_CODIGO;

END VOP_P_ELIMINAR_CICLO;
/
```

1.8.2.13. Procedimiento VOP_P_ELIMINAR_COMI_CODVEND

El siguiente procedimiento elimina los datos de la comisión de un vendedor determinado correspondiente a la tabla CRM_VOP_COMISION_MULTA.

```
CREATE OR REPLACE PROCEDURE VOP_P_ELIMINAR_COMI_CODVEND(
     VI_CODIGO IN CRM_VOP_COMISION_MULTA.CMS_COD_VENDEDOR%TYPE)
IS
BEGIN

  DELETE FROM CRM_VOP_COMISION_MULTA
  WHERE CMS_COD_VENDEDOR  = VI_CODIGO;


END VOP_P_ELIMINAR_COMI_CODVEND;
/
```

1.8.2.14. Procedimiento VOP_P_ELIMINAR_OFERTA

El siguiente procedimiento elimina los datos de la oferta seleccionada correspondiente a la tabla CRM_VOP_OFERTA.

```
CREATE OR REPLACE PROCEDURE VOP_P_ELIMINAR_OFERTA (
                    VI_COD_OFERTA IN CRM_VOP_OFERTA.OFR_COD_OFERTA%TYPE)
IS
BEGIN

  DELETE FROM CRM_VOP_OFERTA
  WHERE OFR_COD_OFERTA  = VI_COD_OFERTA;

END VOP_P_ELIMINAR_OFERTA;
/
```

1.8.2.15. Procedimiento VOP_P_ELIMINAR_PROSPECTO

El siguiente procedimiento elimina lógicamente a un prospecto de la tabla CRM_VOP_PROSPECTO, es decir, que su estadio cambia de A: Activo a I: Inactivo.

```
CREATE OR REPLACE PROCEDURE VOP_P_ELIMINAR_PROSPECTO(
  VI_CODIGO_PROSPECTO IN
CRM_VOP_PROSPECTO.PRO_CODIGO_PROSPECTO%TYPE)
IS
BEGIN

  UPDATE CRM_VOP_PROSPECTO
  SET   PRO_ESTADO = 'I'
  WHERE PRO_CODIGO_PROSPECTO= VI_CODIGO_PROSPECTO;
END VOP_P_ELIMINAR_PROSPECTO;
/
```

1.8.2.16. Procedimiento VOP_P_ELIMINAR_RANGO

El siguiente procedimiento elimina los datos del rango de ventas seleccionado correspondiente a la tabla CRM_VOP_RANGO_VENTAS.

```
CREATE OR REPLACE PROCEDURE VOP_P_ELIMINAR_RANGO (
                    VI_COD_RANGO_VENTAS IN
CRM_VOP_RANGO_VENTAS.RNV_COD_RANGO_VENTAS%TYPE)
IS
BEGIN

  DELETE FROM CRM_VOP_RANGO_VENTAS
  WHERE RNV_COD_RANGO_VENTAS  = VI_COD_RANGO_VENTAS;

END VOP_P_ELIMINAR_RANGO;
/
```

1.8.2.17. Procedimiento VOP_P_ELIMINAR_TELF_CODVEND

El siguiente procedimiento elimina los datos del teléfono del vendedor seleccionado correspondiente a la tabla CRM_VOP_TELEFONO_VENDEDOR.

```
CREATE OR REPLACE PROCEDURE VOP_P_ELIMINAR_TELF_CODVEND(
           VI_CODIGO IN
CRM_VOP_TELEFONO_VENDEDOR.TLV_CODIGO_VENDEDOR%TYPE)
IS
BEGIN

  DELETE FROM CRM_VOP_TELEFONO_VENDEDOR
  WHERE TLV_CODIGO_VENDEDOR  = VI_CODIGO;
END VOP_P_ELIMINAR_TELF_CODVEND;
/

CREATE OR REPLACE PROCEDURE VOP_P_ELIMINAR_TELF_VENDEDOR (
                    VI_CODIGO IN CRM_VOP_TELEFONO_VENDEDOR.TLV_CODIGO%TYPE)
IS
BEGIN

  DELETE FROM CRM_VOP_TELEFONO_VENDEDOR
  WHERE TLV_CODIGO  = VI_CODIGO;

END VOP_P_ELIMINAR_TELF_VENDEDOR;
/
```

1.8.2.18. Procedimiento VOP_P_ELIMINAR_TIPO_VEND

El siguiente procedimiento elimina los datos del tipo de vendedor seleccionado correspondiente a la tabla CRM_VOP_TIPO_VENDEDOR.

```
CREATE OR REPLACE PROCEDURE VOP_P_ELIMINAR_TIPO_VEND (
                    VI_COD_TIPO_VENDEDOR IN
CRM_VOP_TIPO_VENDEDOR.TIV_COD_TIPO_VENDEDOR%TYPE)
IS
BEGIN

  DELETE FROM CRM_VOP_TIPO_VENDEDOR
  WHERE TIV_COD_TIPO_VENDEDOR  = VI_COD_TIPO_VENDEDOR;

END VOP_P_ELIMINAR_TIPO_VEND;
/
```

1.8.2.19. Procedimiento VOP_P_ELIMINAR_VENDEDOR

El siguiente procedimiento elimina los datos del vendedor seleccionado correspondiente a la tabla CRM_VOP_VENDEDOR.

```
CREATE OR REPLACE PROCEDURE VOP_P_ELIMINAR_VENDEDOR (
                    VI_CODIGO IN CRM_VOP_VENDEDOR.VEN_CODIGO%TYPE)
IS
BEGIN

  DELETE FROM CRM_VOP_VENDEDOR
  WHERE VEN_CODIGO  = VI_CODIGO;

END VOP_P_ELIMINAR_VENDEDOR;
/
```

1.8.2.20. Procedimiento VOP_P_ELIMINAR_VISIT_CODVEND

El siguiente procedimiento elimina los datos de la visita asignada a un vendedor seleccionado correspondiente a la tabla CRM_VOP_VISITAS.

```
CREATE OR REPLACE PROCEDURE VOP_P_ELIMINAR_VISIT_CODVEND(
          VI_CODIGO IN CRM_VOP_VISITAS.VST_CODIGO_VENDEDOR%TYPE)
IS
BEGIN

  DELETE FROM CRM_VOP_VISITAS
  WHERE VST_CODIGO_VENDEDOR  = VI_CODIGO;

END VOP_P_ELIMINAR_VISIT_CODVEND;
/
```

1.8.2.21. Procedimiento VOP_P_ELIMINAR_OFERTA_SERV

El siguiente procedimiento elimina los datos de los servicios que pertenecen a una oferta determinada.

```
CREATE OR REPLACE PROCEDURE VOP_P_ELIMINAR_OFERTA_SERV(
  VI_COD_OFERTA    IN CRM_VOP_OFERTA_SERVICIO.OSR_COD_OFERTA%TYPE)
IS
BEGIN

  DELETE FROM CRM_VOP_OFERTA_SERVICIO
  WHERE OSR_COD_OFERTA  = VI_COD_OFERTA;

END VOP_P_ELIMINAR_OFERTA_SERV;
/
```

1.8.2.22. Procedimiento VOP_P_INSERTAR_CANAL

El siguiente procedimiento inserta los datos de un nuevo canal a la tabla CRM_VOP_CANAL donde el código es generado secuencialmente.

```
CREATE OR REPLACE PROCEDURE VOP_P_INSERTAR_CANAL(
  VI_DESCRIPCION   IN CRM_VOP_CANAL.CNL_DESCRIPCIONL%TYPE,
  VI_TIPO_CANAL   IN CRM_VOP_CANAL.CNL_TIPO_CANAL%TYPE,
  VI_PORC_COMISION_CANAL IN
CRM_VOP_CANAL.CNL_PORC_COMISION_CANAL%TYPE)
IS
  VL_CODIGO   CRM_VOP_CANAL.CNL_CODIGO%TYPE;
BEGIN

  SELECT MAX(CNL_CODIGO) INTO VL_CODIGO
  FROM CRM_VOP_CANAL;

  IF VL_CODIGO IS NULL THEN
     VL_CODIGO:=0;
  END IF;

  VL_CODIGO:=VL_CODIGO+1;

  INSERT INTO CRM_VOP_CANAL
  (
   CNL_CODIGO,
   CNL_DESCRIPCIONL,
   CNL_TIPO_CANAL,
   CNL_PORC_COMISION_CANAL
  )
  VALUES (
    TO_CHAR(VL_CODIGO),
    VI_DESCRIPCION,
    VI_TIPO_CANAL,
    VI_PORC_COMISION_CANAL
  );
END VOP_P_INSERTAR_CANAL;
/
```

1.8.2.23. Procedimiento VOP_P_INSERTAR_CICLO

El siguiente procedimiento inserta los datos de un nuevo ciclo de facturación a la tabla CRM_VOP_CICLO_FACT donde el código es generado secuencialmente.

```
CREATE OR REPLACE PROCEDURE VOP_P_INSERTAR_CICLO(
    VI_DESCRIPCION       IN CRM_VOP_CICLO_FACT.CIF_DESCRIPCION%TYPE,
    VI_DIA_INICIO_CICLO  IN CRM_VOP_CICLO_FACT.CIF_DIA_INICIO_CICLO%TYPE,
    VI_DIA_FIN_CICLO     IN CRM_VOP_CICLO_FACT.CIF_DIA_FIN_CICLO%TYPE,
    VI_DIA_INICIO_FACT   IN CRM_VOP_CICLO_FACT.CIF_DIA_INICIO_FACT%TYPE,
    VI_DIA_FIN_FACT      IN CRM_VOP_CICLO_FACT.CIF_DIA_FIN_FACT%TYPE,
    VI_NUM_DIAS_FACT     IN CRM_VOP_CICLO_FACT.CIF_NUM_DIAS_FACT%TYPE)
IS
    VL_CODIGO       CRM_VOP_CICLO_FACT.CIF_CODIGO%TYPE;
BEGIN
    SELECT MAX(CIF_CODIGO) INTO VL_CODIGO
    FROM CRM_VOP_CICLO_FACT;
    IF VL_CODIGO IS NULL THEN
       VL_CODIGO:=0;
    END IF;
    VL_CODIGO:=VL_CODIGO+1;

  INSERT INTO CRM_VOP_CICLO_FACT (
    CIF_CODIGO,
    CIF_DIA_INICIO_CICLO,
    CIF_DIA_FIN_CICLO,
    CIF_DESCRIPCION,
    CIF_DIA_INICIO_FACT,
    CIF_DIA_FIN_FACT,
    CIF_NUM_DIAS_FACT )
  VALUES (
    VL_CODIGO,
    VI_DIA_INICIO_CICLO,
    VI_DIA_FIN_CICLO,
    VI_DESCRIPCION,
    VI_DIA_INICIO_FACT,
    VI_DIA_FIN_FACT,
    VI_NUM_DIAS_FACT );
END VOP_P_INSERTAR_CICLO;/
```

1.8.2.24. Procedimiento VOP_P_INSERTAR_CLIENTE

El siguiente procedimiento inserta los datos de un prospecto en la tabla CRM_SCO_CLIENTES cuando ha sido dado de alta, tomando en consideración el tipo de cliente. Si es cliente es natural se registran los datos correspondientes en la tabla CRM_SCO_CLIENTE_NATURAL y en caso de ser cliente jurídico se insertarán en CRM_SCO_CLIENTE_JURIDICO y el código es generado secuencialmente.

```
CREATE OR REPLACE PROCEDURE VOP_P_INSERTAR_CLIENTE(
  VI_TIPO_CLIENTE    IN CRM_VOP_PROSPECTO.PRO_TIPO_PROSPECTO%TYPE,
  VI_NOMBRES         IN CRM_VOP_PROSPECTO.PRO_NOMBRES%TYPE,
  VI_APELLIDOS       IN CRM_VOP_PROSPECTO.PRO_APELLIDOS%TYPE,
  VI_CEDULA          IN CRM_VOP_PROSPECTO.PRO_CEDULA%TYPE,
  VI_PASAPORTE       IN CRM_VOP_PROSPECTO.PRO_PASAPORTE%TYPE,
  VI_SEXO            IN CRM_VOP_PROSPECTO.PRO_SEXO%TYPE,
  VI_ESTADO_CIVIL    IN CRM_VOP_PROSPECTO.PRO_ESTADO_CIVIL%TYPE,
  VI_FEC_NAC         IN CRM_VOP_PROSPECTO.PRO_FEC_NAC%TYPE,
  VI_ACTIVIDAD       IN CRM_VOP_PROSPECTO.PRO_ACTIVIDAD%TYPE,
  VI_ESTADO          IN CRM_VOP_PROSPECTO.PRO_ESTADO%TYPE,
  VI_NACIONALIDAD    IN CRM_VOP_PROSPECTO.PRO_CODIGO_NAC%TYPE,
  VI_TIPO_CLIENTE_JURIDICO IN
CRM_VOP_PROSPECTO.PRO_TIPO_CLIENTE_JURIDICO%TYPE,
  VI_REP_LEGAL       IN CRM_VOP_PROSPECTO.PRO_REP_LEG%TYPE)
IS
  VL_COD_CLIENTE CRM_VOP_PROSPECTO.PRO_CODIGO_PROSPECTO%TYPE;
  VL_TIPO_PERSONA VARCHAR2(1);
  VL_PASAPORTE CRM_VOP_PROSPECTO.PRO_PASAPORTE%TYPE;
BEGIN

    SELECT MAX(CLI_CODIGO) INTO VL_COD_CLIENTE
    FROM CRM_SCO_CLIENTES;

    IF VL_COD_CLIENTE IS NULL THEN
      VL_COD_CLIENTE:=0;
    END IF;

    VL_COD_CLIENTE:=VL_COD_CLIENTE+1;

    IF VI_TIPO_CLIENTE = 1 THEN
       VL_TIPO_PERSONA:='N';
    ELSE
       VL_TIPO_PERSONA:='J';
    END IF;
```

```
INSERT INTO CRM_SCO_CLIENTES
(
   CLI_CODIGO,
   CLI_PERSONA,
   CLI_ACTIVIDAD,
   CLI_TIPO,
   CLI_ESTADO,
   CLI_NOSERVICIO,
   CLI_CODIGO_NAC
)
VALUES
(
   VL_COD_CLIENTE,
   VL_TIPO_PERSONA,
   VI_ACTIVIDAD,
   VI_TIPO_CLIENTE,
   'A',
   0,
   VI_NACIONALIDAD
);

IF VL_TIPO_PERSONA='N' THEN

  VL_PASAPORTE:=VI_PASAPORTE;
  IF (VL_PASAPORTE IS NULL) OR (VL_PASAPORTE='') THEN
    VL_PASAPORTE:='-';
  END IF;

   INSERT INTO CRM_SCO_CLIENTE_NATURAL
   (
      CLN_ID,
      CLN_NOMBRES,
      CLN_APELLIDOS,
      CLN_CEDULA,
      CLN_PASAPORTE,
      CLN_SEXO,
      CLN_ESTADO_CIVIL,
      CLN_FEC_NAC
   )
   VALUES
   (
      VL_COD_CLIENTE,
      VI_NOMBRES,
      VI_APELLIDOS,
      VI_CEDULA,
      VL_PASAPORTE,
      VI_SEXO,
      VI_ESTADO_CIVIL,
      VI_FEC_NAC
   );
ELSE
```

```
INSERT INTO CRM_SCO_CLIENTE_JURIDICO
       (
          CLJ_ID,
          CLJ_TIPO,
          CLJ_RUC,
          CLJ_RAZON,
          CLJ_NOMB_COM,
          CLJ_REP_LEG
       )
       VALUES
       (
          VL_COD_CLIENTE,
          VI_TIPO_CLIENTE_JURIDICO,
          VI_CEDULA,
          VI_APELLIDOS,
          VI_NOMBRES,
          VI_REP_LEGAL
        );
    END IF;

END VOP_P_INSERTAR_CLIENTE;
/
```

1.8.2.25. Procedimiento VOP_P_INSERTAR_DET_SOLICITUD

El siguiente procedimiento inserta los datos del detalle de solicitud a la tabla CRM_VOP_DET_SOLICITUD donde el código es generado secuencialmente.

```
CREATE OR REPLACE PROCEDURE VOP_P_INSERTAR_DET_SOLICITUD(
  VI_NUM_SOLICITUD IN CRM_VOP_DET_SOLICITUD.DSL_NUM_SOLICITUD%TYPE,
  VI_COD_SERVICIO  IN CRM_VOP_DET_SOLICITUD.DSL_COD_SERVICIO%TYPE,
  VI_CANTIDAD_SERVICIO IN
CRM_VOP_DET_SOLICITUD.DSL_CANTIDAD_SERVICIO%TYPE,
  VI_VALOR_SERVICIO IN CRM_VOP_DET_SOLICITUD.DSL_VALOR_SERVICIO%TYPE
)
IS
  VL_NUM_ITEM CRM_VOP_DET_SOLICITUD.DSL_NUM_ITEM%TYPE;
BEGIN
 VL_NUM_ITEM:=0;

 SELECT NVL(MAX(DSL_NUM_ITEM),0) INTO VL_NUM_ITEM
 FROM CRM_VOP_DET_SOLICITUD
 WHERE DSL_NUM_SOLICITUD=VI_NUM_SOLICITUD;

 IF VL_NUM_ITEM IS NULL THEN
   VL_NUM_ITEM:=0;
 END IF;

 VL_NUM_ITEM:=VL_NUM_ITEM+1;
 INSERT INTO CRM_VOP_DET_SOLICITUD
 (
    DSL_NUM_SOLICITUD,
    DSL_NUM_ITEM,
    DSL_COD_SERVICIO,
    DSL_CANTIDAD_SERVICIO,
    DSL_VALOR_SERVICIO
 )
 VALUES (
    VI_NUM_SOLICITUD,
    TO_CHAR(VL_NUM_ITEM),
    VI_COD_SERVICIO,
    VI_CANTIDAD_SERVICIO,
    VI_VALOR_SERVICIO
 );
END VOP_P_INSERTAR_DET_SOLICITUD;
/
```

1.8.2.26. Procedimiento VOP_P_INSERTAR_PROSPECTO

El siguiente procedimiento inserta los datos de un nuevo prospecto a la tabla CRM_VOP_PROSPECTO donde el código se genera secuencialmente, y el tipo de persona ingresada puede ser natural o jurídica.

Sólo para las personas naturales se registra sexo, fecha de nacimiento, estado civil, y pasaporte (éste último no es obligatorio).

En el caso de personas jurídicas, se recibe datos de tipo de cliente jurídico y representante legal.

Los campos Nombres, Apellidos y Cédula cumplen doble funcionalidad; para las personas naturales los datos que se registran son los indicados por sus nombres, pero en el caso de personas jurídicas se registra el nombre comercial, razón social y número de RUC, respectivamente.

```
CREATE OR REPLACE PROCEDURE VOP_P_INSERTAR_PROSPECTO(
  VI_TIPO_PROSPECTO   IN CRM_VOP_PROSPECTO.PRO_TIPO_PROSPECTO%TYPE,
  VI_NOMBRES          IN CRM_VOP_PROSPECTO.PRO_NOMBRES%TYPE,

  VI_APELLIDOS        IN CRM_VOP_PROSPECTO.PRO_APELLIDOS%TYPE,
  VI_CEDULA           IN CRM_VOP_PROSPECTO.PRO_CEDULA%TYPE,

  VI_PASAPORTE        IN CRM_VOP_PROSPECTO.PRO_PASAPORTE%TYPE,
  VI_SEXO             IN CRM_VOP_PROSPECTO.PRO_SEXO%TYPE,
  VI_ESTADO_CIVIL     IN CRM_VOP_PROSPECTO.PRO_ESTADO_CIVIL%TYPE,
  VI_FECHA_NAC        IN CRM_VOP_PROSPECTO.PRO_FEC_NAC%TYPE,

  VI_ACTIVIDAD        IN CRM_VOP_PROSPECTO.PRO_ACTIVIDAD%TYPE,
  VI_NACIONALIDAD     IN CRM_VOP_PROSPECTO.PRO_CODIGO_NAC%TYPE,
  VI_ESTADO           IN CRM_VOP_PROSPECTO.PRO_ESTADO%TYPE,
  VI_TIPO_CLIENTE_JURIDICO   IN
CRM_VOP_PROSPECTO.PRO_TIPO_CLIENTE_JURIDICO%TYPE,
  VI_REP_LEGAL       IN CRM_VOP_PROSPECTO.PRO_REP_LEG%TYPE)
IS
  VL_CODIGO_PROSPECTO
CRM_VOP_PROSPECTO.PRO_CODIGO_PROSPECTO%TYPE;
BEGIN

  SELECT MAX(PRO_CODIGO_PROSPECTO) INTO VL_CODIGO_PROSPECTO
  FROM CRM_VOP_PROSPECTO;
```

```
  IF VL_CODIGO_PROSPECTO IS NULL THEN
     VL_CODIGO_PROSPECTO:=0;
  END IF;

  VL_CODIGO_PROSPECTO:=VL_CODIGO_PROSPECTO+1;

INSERT INTO CRM_VOP_PROSPECTO
 (
  PRO_CODIGO_PROSPECTO,
  PRO_TIPO_PROSPECTO,
  PRO_NOMBRES,
  PRO_APELLIDOS,
  PRO_CEDULA,
  PRO_PASAPORTE,
  PRO_SEXO,
  PRO_ESTADO_CIVIL,
  PRO_FEC_NAC,
  PRO_ACTIVIDAD,
  PRO_CODIGO_NAC,
  PRO_ESTADO,
  PRO_TIPO_CLIENTE_JURIDICO,
  PRO_REP_LEG
 )
 VALUES (
   VL_CODIGO_PROSPECTO,
   VI_TIPO_PROSPECTO,
   VI_NOMBRES,
   VI_APELLIDOS,
   NVL2(VI_CEDULA,VI_CEDULA,''),
   NVL2(VI_PASAPORTE,VI_PASAPORTE,''),
   NVL2(VI_SEXO,VI_SEXO,''),
   NVL2(VI_ESTADO_CIVIL,VI_ESTADO_CIVIL,''),
   NVL2(VI_FECHA_NAC,VI_FECHA_NAC,''),
   VI_ACTIVIDAD,
   VI_NACIONALIDAD,
   VI_ESTADO,
   NVL2(VI_TIPO_CLIENTE_JURIDICO,VI_TIPO_CLIENTE_JURIDICO,''),
   NVL2(VI_REP_LEGAL,VI_REP_LEGAL,'')
 );
END VOP_P_INSERTAR_PROSPECTO;
/
```

1.8.2.27.Procedimiento VOP_P_INSERTAR_RANGO_VENTAS

El siguiente procedimiento inserta los datos de un nuevo rango de ventas a la tabla CRM_VOP_RANGO_VENTAS donde el código es generado secuencialmente.

```
CREATE OR REPLACE PROCEDURE VOP_P_INSERTAR_RANGO_VENTAS(
    VI_LIMITE_INFERIOR      IN CRM_VOP_RANGO_VENTAS.RNV_LIMITE_INFERIOR%TYPE,
    VI_LIMITE_SUPERIOR      IN CRM_VOP_RANGO_VENTAS.RNV_LIMITE_SUPERIOR%TYPE,
    VI_PORCENTAJE_COMISION  IN
CRM_VOP_RANGO_VENTAS.RNV_PORCENTAJE_COMISION%TYPE,
    VI_FECHA_INICIAL_RANGO  IN
CRM_VOP_RANGO_VENTAS.RNV_FECHA_INICIAL_RANGO%TYPE,
    VI_FECHA_FINAL_RANGO    IN
CRM_VOP_RANGO_VENTAS.RNV_FECHA_FINAL_RANGO%TYPE,
    VI_TIPO_META            IN CRM_VOP_RANGO_VENTAS.RNV_TIPO_META%TYPE,
    VI_DESCRIPCION          IN CRM_VOP_RANGO_VENTAS.RNV_DESCRIPCION%TYPE)
IS
    VL_COD_RANGO_VENTAS  CRM_VOP_RANGO_VENTAS.RNV_COD_RANGO_VENTAS%TYPE;
BEGIN
    SELECT MAX(RNV_COD_RANGO_VENTAS) INTO VL_COD_RANGO_VENTAS
    FROM CRM_VOP_RANGO_VENTAS;

    IF VL_COD_RANGO_VENTAS IS NULL THEN
       VL_COD_RANGO_VENTAS:=0;
    END IF;
    VL_COD_RANGO_VENTAS:=VL_COD_RANGO_VENTAS+1;

   INSERT INTO CRM_VOP_RANGO_VENTAS
   (
     RNV_COD_RANGO_VENTAS,
     RNV_LIMITE_INFERIOR,
     RNV_LIMITE_SUPERIOR,
     RNV_PORCENTAJE_COMISION,
     RNV_FECHA_INICIAL_RANGO,
     RNV_FECHA_FINAL_RANGO,
     RNV_TIPO_META,
     RNV_DESCRIPCION    )
   VALUES (
     TO_CHAR(VL_COD_RANGO_VENTAS),
     VI_LIMITE_INFERIOR,
     VI_LIMITE_SUPERIOR,
     VI_PORCENTAJE_COMISION,
     VI_FECHA_INICIAL_RANGO,
     VI_FECHA_FINAL_RANGO,
     VI_TIPO_META,
     VI_DESCRIPCION
   );

END VOP_P_INSERTAR_RANGO_VENTAS;
/
```

1.8.2.28. Procedimiento VOP_P_INSERTAR_TELF_VEND

El siguiente procedimiento inserta los datos de un nuevo teléfono del vendedor seleccionado a la tabla CRM_VOP_TELEFONO_VENDEDOR donde el código es generado secuencialmente.

```
CREATE OR REPLACE PROCEDURE VOP_P_INSERTAR_TELF_VEND(
    VI_DESCRIPCION          IN
CRM_VOP_TELEFONO_VENDEDOR.TLV_DESCRIPCION%TYPE,
    VI_CODIGO_VENDEDOR      IN
CRM_VOP_TELEFONO_VENDEDOR.TLV_CODIGO_VENDEDOR%TYPE,
    VI_CODIGO_TIPO_TLF      IN
CRM_VOP_TELEFONO_VENDEDOR.TLV_CODIGO_TIPO_TLF%TYPE
)

IS
    VL_CODIGO   CRM_VOP_TELEFONO_VENDEDOR.TLV_CODIGO%TYPE;
BEGIN

    SELECT MAX(TLV_CODIGO) INTO VL_CODIGO
    FROM CRM_VOP_TELEFONO_VENDEDOR;

    IF VL_CODIGO IS NULL THEN
      VL_CODIGO:=0;
    END IF;

    VL_CODIGO:=VL_CODIGO+1;

  INSERT INTO CRM_VOP_TELEFONO_VENDEDOR
  (
    TLV_CODIGO,
    TLV_DESCRIPCION,
    TLV_CODIGO_VENDEDOR,
    TLV_CODIGO_TIPO_TLF
  )
  VALUES (
    VL_CODIGO,
    VI_DESCRIPCION,
    VI_CODIGO_VENDEDOR,
    VI_CODIGO_TIPO_TLF
  );
END VOP_P_INSERTAR_TELF_VEND;
/
```

1.8.2.29. Procedimiento VOP_P_INSERTAR_TIPO_VENDEDOR

El siguiente procedimiento inserta los datos de un nuevo tipo de vendedor a la tabla CRM_VOP_TIPO_VENDEDOR donde el código es generado secuencialmente.

```
CREATE OR REPLACE PROCEDURE VOP_P_INSERTAR_TIPO_VENDEDOR(
    VI_DESCRIPCION_TIPO_VENDEDOR IN
CRM_VOP_TIPO_VENDEDOR.TIV_DESCRIPCION_TIPO_VENDEDOR%TYPE
)
IS
  VL_COD_TIPO_VENDEDOR
CRM_VOP_TIPO_VENDEDOR.TIV_COD_TIPO_VENDEDOR%TYPE;
BEGIN

    SELECT MAX(TIV_COD_TIPO_VENDEDOR) INTO VL_COD_TIPO_VENDEDOR
    FROM CRM_VOP_TIPO_VENDEDOR;

    IF VL_COD_TIPO_VENDEDOR IS NULL THEN
      VL_COD_TIPO_VENDEDOR:=0;
    END IF;

    VL_COD_TIPO_VENDEDOR:=VL_COD_TIPO_VENDEDOR+1;

  INSERT INTO CRM_VOP_TIPO_VENDEDOR
  (
    TIV_COD_TIPO_VENDEDOR,
    TIV_DESCRIPCION_TIPO_VENDEDOR
  )
  VALUES (
    TO_CHAR(VL_COD_TIPO_VENDEDOR),
    VI_DESCRIPCION_TIPO_VENDEDOR
  );

END VOP_P_INSERTAR_TIPO_VENDEDOR;
/
```

1.8.2.30. Procedimiento VOP_P_INSERTAR_VENDEDOR

El siguiente procedimiento inserta los datos de un nuevo vendedor a la tabla CRM_VOP_VENDEDOR donde el código del vendedor es igual al código que tiene en la tabla CRM_ADM_EMPLEADOS.

```
CREATE OR REPLACE PROCEDURE VOP_P_INSERTAR_VENDEDOR(
    VI_CODIGO            IN CRM_VOP_VENDEDOR.VEN_CODIGO%TYPE,
    VI_COD_TIPO_VENDEDOR  IN
CRM_VOP_VENDEDOR.VEN_COD_TIPO_VENDEDOR%TYPE,
    VI_COD_RANGO_VENTAS   IN
CRM_VOP_VENDEDOR.VEN_COD_RANGO_VENTAS%TYPE,
    VI_COD_CANAL    IN CRM_VOP_VENDEDOR.VEN_COD_CANAL%TYPE,
    VI_COMISIONES   IN CRM_VOP_VENDEDOR.VEN_COMISIONES%TYPE,
    VI_ESTADO       IN CRM_VOP_VENDEDOR.VEN_ESTADO%TYPE
)
IS
    --VL_CODIGO  CRM_VOP_VENDEDOR.VEN_CODIGO%TYPE;

 BEGIN
  INSERT INTO CRM_VOP_VENDEDOR
  (
    VEN_CODIGO,
    VEN_COD_TIPO_VENDEDOR,
    VEN_COD_RANGO_VENTAS,
    VEN_COD_CANAL,
    VEN_COMISIONES,
    VEN_ESTADO
  )
  VALUES (
    --TO_CHAR(VL_CODIGO),
    VI_CODIGO,
    VI_COD_TIPO_VENDEDOR,
    VI_COD_RANGO_VENTAS,
    VI_COD_CANAL,
    VI_COMISIONES,
    VI_ESTADO
  );

END VOP_P_INSERTAR_VENDEDOR;
/
```

1.8.2.31. Procedimiento VOP_P_INSERTAR_VISITAS

El siguiente procedimiento inserta los datos de una nueva visita para un vendedor seleccionado a la tabla CRM_VOP_VISITAS donde los códigos de visita y control son generados secuencialmente, según el caso.
El parámetro vi_operación recibe 'N' si desde el programa fue invocado el botón "Nueva visita" o 'P' si se escogió "Próxima visita".
En el caso de una nueva visita, se genera un nuevo código para id_visita y el campo vst_control inicia con 1.
Cuando se registra una próxima visita, significa que a partir de un id_visita, se va a insertar otro registro con el mismo número de id_visita pero se incrementa vst_control. Esto se realiza con la finalidad de registrar información de una visita adicional realizada por el mismo motivo anterior, en caso de no haberse finiquitado la negociación.

```
CREATE OR REPLACE PROCEDURE VOP_P_INSERTAR_VISITAS(
    VI_ID_VISITA        IN CRM_VOP_VISITAS.VST_ID_VISITA%TYPE,
    VI_CODIGO_CLIENTE   IN CRM_VOP_VISITAS.VST_CODIGO_CLIENTE%TYPE,
    VI_CODIGO_VENDEDOR  IN CRM_VOP_VISITAS.VST_CODIGO_VENDEDOR%TYPE,
    VI_CODIGO_CANAL     IN CRM_VOP_VISITAS.VST_CODIGO_CANAL%TYPE,
    VI_FECHA_ASIGNACION IN CRM_VOP_VISITAS.VST_FECHA_ASIGNACION%TYPE,
    VI_ESTADO           IN CRM_VOP_VISITAS.VST_ESTADO%TYPE,
    VI_OBSERVACION      IN CRM_VOP_VISITAS.VST_OBSERVACION%TYPE,
    VI_OPERACION        IN VARCHAR2 --'N' o 'P'
)
IS
    VL_CONTROL      CRM_VOP_VISITAS.VST_CONTROL%TYPE;
    VL_ID_VISITA    CRM_VOP_VISITAS.VST_ID_VISITA%TYPE;
BEGIN

    IF VI_OPERACION='N' THEN
      SELECT MAX(VST_ID_VISITA) INTO VL_ID_VISITA
      FROM CRM_VOP_VISITAS;

      IF VL_ID_VISITA IS NULL THEN
        VL_ID_VISITA:=0;
      END IF;
      VL_ID_VISITA:=VL_ID_VISITA+1;
      VL_CONTROL:=1;
    END IF;
```

```
    IF VI_OPERACION='P' THEN
      VL_ID_VISITA:=VI_ID_VISITA;
      SELECT MAX(VST_CONTROL) INTO VL_CONTROL
      FROM CRM_VOP_VISITAS
      WHERE VST_ID_VISITA=VL_ID_VISITA;

      IF VL_CONTROL IS NULL THEN
        VL_CONTROL:=0;
      END IF;
      VL_CONTROL:=VL_CONTROL+1;
    END IF;
    INSERT INTO CRM_VOP_VISITAS
    (
        VST_CONTROL,
        VST_ID_VISITA,
        VST_CODIGO_CLIENTE,
        VST_CODIGO_VENDEDOR,
        VST_CODIGO_CANAL,
        VST_FECHA_ASIGNACION,
        --VST_FECHA_VISITA,
        VST_ESTADO,
        VST_OBSERVACION
    )
    VALUES (
        VL_CONTROL,
        VL_ID_VISITA,
        VI_CODIGO_CLIENTE,
        VI_CODIGO_VENDEDOR,
        VI_CODIGO_CANAL,
        VI_FECHA_ASIGNACION,
        --VI_FECHA_VISITA,
        VI_ESTADO,
        VI_OBSERVACION
    );

END VOP_P_INSERTAR_VISITAS;
/
```

1.8.2.32. Procedimiento VOP_P_REGISTRAR_PAGO

El siguiente procedimiento permite registrar el valor y la fecha de las facturas que son ingresadas al sistema mediante la carga de un archivo plano proveniente de una entidad bancaria.

```
CREATE OR REPLACE PROCEDURE VOP_P_REGISTRAR_PAGO(
        VI_NUM_FACTURA IN CRM_VOP_CAB_FACTURA.CFAC_NUM_FACTURA%TYPE,
        VI_COD_CLIENTE IN CRM_VOP_CAB_FACTURA.CFAC_COD_CLIENTE%TYPE,
        VI_FECHA_PAGO  IN CRM_VOP_CAB_FACTURA.CFAC_FECHA_PAGO%TYPE)
        IS
BEGIN

  UPDATE CRM_VOP_CAB_FACTURA
    SET  CFAC_ESTADO = 'C',
        CFAC_VALOR_ABONO =  CFAC_TOTAL,
        CFAC_FECHA_PAGO = VI_FECHA_PAGO
    WHERE CFAC_NUM_FACTURA = VI_NUM_FACTURA
    AND   CFAC_COD_CLIENTE = VI_COD_CLIENTE;

END VOP_P_REGISTRAR_PAGO;
/
```

1.8.2.33. Procedimiento VOP_P_INSERTAR_OFERTA_SERVICIO

El siguiente procedimiento permite registrar los servicios que pertenecen a una oferta, utilizando el código que tiene en la tabla CRM_VOP_OFERTA, y generando códigos para los números de item.

```
CREATE OR REPLACE PROCEDURE VOP_P_INSERTAR_OFERTA_SERVICIO(
    VI_COD_OFERTA    IN CRM_VOP_OFERTA_SERVICIO.OSR_COD_OFERTA%TYPE,
    VI_COD_ITEM      IN CRM_VOP_OFERTA_SERVICIO.OSR_COD_ITEM%TYPE,
    VI_PRECIO_ITEM   IN CRM_VOP_OFERTA_SERVICIO.OSR_PRECIO_ITEM%TYPE
)
IS
VL_NUM_ITEM CRM_VOP_OFERTA_SERVICIO.OSR_NUM_ITEM%TYPE;
BEGIN
  VL_NUM_ITEM := 0;

  SELECT MAX(OFR.OSR_NUM_ITEM) INTO VL_NUM_ITEM
  FROM CRM_VOP_OFERTA_SERVICIO OFR;

  IF VL_NUM_ITEM IS NULL THEN
    VL_NUM_ITEM := 0;
  END IF;

  VL_NUM_ITEM:=VL_NUM_ITEM+1;

  INSERT INTO CRM_VOP_OFERTA_SERVICIO
  (
    OSR_NUM_ITEM,
    OSR_COD_OFERTA,
    OSR_COD_ITEM,
    OSR_PRECIO_ITEM
  )
  VALUES (
    VL_NUM_ITEM,
    VI_COD_OFERTA,
    VI_COD_ITEM,
    VI_PRECIO_ITEM
  );

END VOP_P_INSERTAR_OFERTA_SERVICIO;
```

1.8.3 Paquetes almacenados

1.8.3.1 Paquete para cálculo de comisiones

```
CREATE OR REPLACE PACKAGE VOP_K_COMISIONAR IS

 FUNCTION VOP_F_NUM_SECUENCIAL_CMS(VO_CODIGO_CMS OUT
CRM_VOP_COMISION_MULTA.CMS_COD_COMISIONES%TYPE)
 RETURN CRM_VOP_COMISION_MULTA.CMS_COD_COMISIONES%TYPE;

 FUNCTION VOP_F_CALCULAR_COM_SERV(
      VI_CODIGO_VENDEDOR  IN  CRM_VOP_VENDEDOR.VEN_CODIGO%TYPE,
      VI_VALOR_MIN_SERV   IN
CRM_ADM_PARAMETROS.PRM_VALOR_SERV_MIN_VIP%TYPE,
      VO_VALOR_COMISION   OUT CRM_VOP_VENTA.VTS_VALOR%TYPE)
 RETURN CRM_VOP_VENTA.VTS_VALOR%TYPE;

 FUNCTION VOP_F_CALCULAR_COM_CANAL(
       VI_CODIGO_VENDEDOR  IN  CRM_VOP_VENDEDOR.VEN_CODIGO%TYPE,
       VI_CODIGO_CANAL     IN  CRM_VOP_CANAL.CNL_CODIGO%TYPE,
       VO_VALOR_COMISION   OUT CRM_VOP_VENTA.VTS_VALOR%TYPE)
 RETURN CRM_VOP_VENTA.VTS_VALOR%TYPE;

 FUNCTION VOP_F_CALCULAR_COM_ROT(
      VI_CODIGO_VENDEDOR  IN  CRM_VOP_VENDEDOR.VEN_CODIGO%TYPE,
      VO_VALOR_COMISION   OUT CRM_VOP_VENTA.VTS_VALOR%TYPE)
 RETURN CRM_VOP_VENTA.VTS_VALOR%TYPE;

 FUNCTION VOP_F_CALCULAR_COM_RV(
       VI_COD_VEND          IN CRM_VOP_VENDEDOR.VEN_CODIGO%TYPE,
       VI_TIPO_CALCULO      IN VARCHAR2,
       VO_COMISION_RV       OUT CRM_VOP_VENDEDOR.VEN_COMISIONES%TYPE)
 RETURN CRM_VOP_VENDEDOR.VEN_COMISIONES%TYPE;

 FUNCTION VOP_F_CANAL_VENDEDOR(
       VI_COD_VEND         IN CRM_VOP_VENDEDOR.VEN_CODIGO%TYPE,
       VO_COD_CANAL        OUT CRM_VOP_VENDEDOR.VEN_COD_CANAL%TYPE)
 RETURN CRM_VOP_VENDEDOR.VEN_COD_CANAL%TYPE;
```

```
FUNCTION VOP_F_CALCULAR_MULTA(
        VI_CODIGO       IN  CRM_VOP_VENDEDOR.VEN_CODIGO%TYPE,
        VO_VALOR_MULTA      OUT CRM_VOP_VENTA.VTS_VALOR%TYPE)
  RETURN CRM_VOP_VENTA.VTS_VALOR%TYPE;

  PROCEDURE VOP_P_INSERTAR_COMISION_MULTA(
    VI_COD_COMISIONES   IN
CRM_VOP_COMISION_MULTA.CMS_COD_COMISIONES%TYPE,
    VI_COD_VENDEDOR     IN CRM_VOP_COMISION_MULTA.CMS_COD_VENDEDOR%TYPE,
    VI_DESCRIPCION      IN CRM_VOP_COMISION_MULTA.CMS_DESCRIPCION%TYPE,
    VI_VALOR_COMISIONES IN
CRM_VOP_COMISION_MULTA.CMS_VALOR_COMISIONES%TYPE,
    VI_FECHA_COMISIONES IN
CRM_VOP_COMISION_MULTA.CMS_FECHA_COMISIONES%TYPE,
    VI_VALOR_MULTA      IN CRM_VOP_COMISION_MULTA.CMS_VALOR_MULTA%TYPE,
    VI_VALOR_COM_CANAL  IN
CRM_VOP_COMISION_MULTA.CMS_VALOR_COM_CANAL%TYPE,
    VI_VALOR_COM_SERVICIO   IN
CRM_VOP_COMISION_MULTA.CMS_VALOR_COM_SERVICIO%TYPE,
    VI_VALOR_COM_ROTACION   IN
CRM_VOP_COMISION_MULTA.CMS_VALOR_COM_ROTACION%TYPE,
    VI_VALOR_COM_INCENTIVO  IN
CRM_VOP_COMISION_MULTA.CMS_VALOR_COM_INCENTIVO%TYPE
  );

  PROCEDURE VOP_P_OBTENER_COMISION(
         VI_COD_VENDEDOR        IN CRM_VOP_VENDEDOR.VEN_CODIGO%TYPE,
         VI_COM_ROTACION        IN VARCHAR2,
         VI_COM_SERVICIO        IN VARCHAR2,
         VI_COM_INCENTIVO       IN VARCHAR2,
         VI_COM_CANAL           IN VARCHAR2,
         VI_COM_RANGO_VTAS      IN VARCHAR2);
end VOP_K_COMISIONAR;
/
```

```
CREATE OR REPLACE PACKAGE BODY VOP_K_COMISIONAR IS
  -- FUNCTION AND PROCEDURE IMPLEMENTATIONS
  /*--- FUNCION OBTENER SECUENCIAL COMISIONES-MULTAS ---*/
  FUNCTION VOP_F_NUM_SECUENCIAL_CMS(VO_CODIGO_CMS OUT
CRM_VOP_COMISION_MULTA.CMS_COD_COMISIONES%TYPE)
  RETURN CRM_VOP_COMISION_MULTA.CMS_COD_COMISIONES%TYPE IS
  VL_CODIGO_CMS NUMBER;
  BEGIN
     VL_CODIGO_CMS:=0;
     SELECT NVL(MAX(CMS_COD_COMISIONES),'0') INTO VL_CODIGO_CMS
     FROM CRM_VOP_COMISION_MULTA;
     IF VL_CODIGO_CMS IS NULL THEN
       VL_CODIGO_CMS:=0;
     END IF;
     VL_CODIGO_CMS:=VL_CODIGO_CMS+1;
     VO_CODIGO_CMS:=TO_CHAR(VL_CODIGO_CMS);
     RETURN VO_CODIGO_CMS;
  END;

  /*--- FUNCION COMISION SERVICIO ---*/
  FUNCTION VOP_F_CALCULAR_COM_SERV(
        VI_CODIGO_VENDEDOR  IN  CRM_VOP_VENDEDOR.VEN_CODIGO%TYPE,
        VI_VALOR_MIN_SERV   IN
CRM_ADM_PARAMETROS.PRM_VALOR_SERV_MIN_VIP%TYPE,
        VO_VALOR_COMISION   OUT CRM_VOP_VENTA.VTS_VALOR%TYPE)
        RETURN CRM_VOP_VENTA.VTS_VALOR%TYPE AS

        VL_PORC_COM
CRM_ADM_PARAMETROS.PRM_PORC_COMISION_SERVICIO%TYPE;
        VL_TOTAL_VTA   CRM_VOP_VENTA.VTS_VALOR%TYPE;
        VL_FECHA1      DATE;
        VL_FECHA2      DATE;
        VL_DIA_COMISIONA NUMBER(2);
        VL_DIA_INICIO NUMBER(2);
        VL_MES_INICIO NUMBER(2);
        VL_MES        VARCHAR2(3);
        VL_ANIO_INICIO NUMBER(4);
```

```
BEGIN

    SELECT PRM_PORC_COMISION_SERVICIO
    INTO   VL_PORC_COM
    FROM   CRM_ADM_PARAMETROS;
    SELECT PRM_DIA_COMISIONA
    INTO VL_DIA_COMISIONA
    FROM CRM_ADM_PARAMETROS;

    VL_FECHA2 := TO_DATE(TO_CHAR(VL_DIA_COMISIONA) || '/' ||
TO_CHAR(SYSDATE,'MON') || '/' || TO_CHAR(SYSDATE,'YYYY'));
    SELECT TO_NUMBER(TO_CHAR(VL_FECHA2+1,'DD')),
         TO_NUMBER(TO_CHAR(VL_FECHA2,'MM')),
         TO_NUMBER(TO_CHAR(VL_FECHA2,'YYYY'))
    INTO VL_DIA_INICIO,VL_MES_INICIO,VL_ANIO_INICIO
    FROM DUAL;
    IF (VL_MES_INICIO-1 = 0) THEN
      VL_MES_INICIO:=12;
      VL_ANIO_INICIO:=VL_ANIO_INICIO-1;
    ELSE
      VL_MES_INICIO:=VL_MES_INICIO-1;
    END IF;
    VL_MES:=
    CASE
       WHEN VL_MES_INICIO=1 THEN 'JAN'
       WHEN VL_MES_INICIO=2 THEN 'FEB'
       WHEN VL_MES_INICIO=3 THEN 'MAR'
       WHEN VL_MES_INICIO=4 THEN 'APR'
       WHEN VL_MES_INICIO=5 THEN 'MAY'
       WHEN VL_MES_INICIO=6 THEN 'JUN'
       WHEN VL_MES_INICIO=7 THEN 'JUL'
       WHEN VL_MES_INICIO=8 THEN 'AUG'
       WHEN VL_MES_INICIO=9 THEN 'SEP'
       WHEN VL_MES_INICIO=10 THEN 'OCT'
       WHEN VL_MES_INICIO=11 THEN 'NOV'
       WHEN VL_MES_INICIO=12 THEN 'DEC'
    END;
```

```
   VL_FECHA1 :=
TO_DATE(TO_CHAR(VL_DIA_INICIO)||'/'||VL_MES||'/'||TO_CHAR(VL_ANIO_INICIO));

    SELECT SUM(SC.SEC_PRECIO_SERV)
    INTO   VL_TOTAL_VTA
    FROM   CRM_VOP_VENDEDOR VN, CRM_VOP_CONTRATO_CLIENTE CC,
           CRM_VOP_SERV_CONT SC
    WHERE  VN.VEN_CODIGO      = VI_CODIGO_VENDEDOR
    AND    VN.VEN_CODIGO      = CC.CTC_COD_VENDEDOR
    AND    CC.CTC_CODCONTRATO = SC.SEC_COD_CONTRATO
    AND    CC.CTC_CODCLIENTE = SC.SEC_CODCLIENTE
    AND    SC.SEC_PRECIO_SERV >= VI_VALOR_MIN_SERV
    AND    CC.CTC_FEC_INICIO BETWEEN VL_FECHA1 AND VL_FECHA2
    AND    CC.CTC_FECHA_RECISION IS NULL;

    VO_VALOR_COMISION := VL_TOTAL_VTA * (VL_PORC_COM /100);

    RETURN(VO_VALOR_COMISION);
END VOP_F_CALCULAR_COM_SERV;

/*--- FUNCION COMISION CANAL ---*/
FUNCTION VOP_F_CALCULAR_COM_CANAL(
       VI_CODIGO_VENDEDOR  IN  CRM_VOP_VENDEDOR.VEN_CODIGO%TYPE,
       VI_CODIGO_CANAL     IN  CRM_VOP_CANAL.CNL_CODIGO%TYPE,
       VO_VALOR_COMISION   OUT CRM_VOP_VENTA.VTS_VALOR%TYPE)
RETURN CRM_VOP_VENTA.VTS_VALOR%TYPE
AS
     VL_PORC_COM    CRM_VOP_CANAL.CNL_PORC_COMISION_CANAL%TYPE;
     VL_TOTAL_VTA   CRM_VOP_VENTA.VTS_VALOR%TYPE;
     VL_FECHA1      DATE;
     VL_FECHA2      DATE;
     VL_DIA_COMISIONA NUMBER(2);
     VL_DIA_INICIO NUMBER(2);
     VL_MES_INICIO NUMBER(2);
     VL_MES        VARCHAR2(3);
     VL_ANIO_INICIO NUMBER(4);
```

```
BEGIN
      SELECT CNL_PORC_COMISION_CANAL
      INTO   VL_PORC_COM
      FROM   CRM_VOP_CANAL
      WHERE  CNL_CODIGO   = VI_CODIGO_CANAL;

      SELECT PRM_DIA_COMISIONA
      INTO VL_DIA_COMISIONA
      FROM CRM_ADM_PARAMETROS;

      VL_FECHA2 := TO_DATE(TO_CHAR(VL_DIA_COMISIONA) || '/' ||
TO_CHAR(SYSDATE,'MON') || '/' || TO_CHAR(SYSDATE,'YYYY'));
      --VL_FECHA1 := VL_FECHA2 - 28;
      SELECT TO_NUMBER(TO_CHAR(VL_FECHA2+1,'DD')),
            TO_NUMBER(TO_CHAR(VL_FECHA2,'MM')),
            TO_NUMBER(TO_CHAR(VL_FECHA2,'YYYY'))
      INTO VL_DIA_INICIO,VL_MES_INICIO,VL_ANIO_INICIO
      FROM DUAL;

      IF (VL_MES_INICIO-1 = 0) THEN
        VL_MES_INICIO:=12;
        VL_ANIO_INICIO:=VL_ANIO_INICIO-1;
      ELSE
        VL_MES_INICIO:=VL_MES_INICIO-1;
      END IF;

      VL_MES:=
      CASE
          WHEN VL_MES_INICIO=1 THEN 'JAN'
          WHEN VL_MES_INICIO=2 THEN 'FEB'
          WHEN VL_MES_INICIO=3 THEN 'MAR'
          WHEN VL_MES_INICIO=4 THEN 'APR'
          WHEN VL_MES_INICIO=5 THEN 'MAY'
          WHEN VL_MES_INICIO=6 THEN 'JUN'
          WHEN VL_MES_INICIO=7 THEN 'JUL'
          WHEN VL_MES_INICIO=8 THEN 'AUG'
          WHEN VL_MES_INICIO=9 THEN 'SEP'
```

```
 WHEN VL_MES_INICIO=10 THEN 'OCT'
         WHEN VL_MES_INICIO=11 THEN 'NOV'
         WHEN VL_MES_INICIO=12 THEN 'DEC'
      END;

      VL_FECHA1 := TO_CHAR(VL_DIA_INICIO)||'/'||VL_MES||'/'||TO_CHAR(VL_ANIO_INICIO);
      SELECT SUM( VT.VTS_VALOR )
      INTO   VL_TOTAL_VTA
      FROM   CRM_VOP_VENDEDOR VN, CRM_VOP_CONTRATO_CLIENTE CC,
CRM_VOP_VENTA VT, CRM_VOP_CANAL CN
      WHERE  VN.VEN_CODIGO        = VI_CODIGO_VENDEDOR
      AND    VN.VEN_CODIGO        = CC.CTC_COD_VENDEDOR
      AND    CC.CTC_CODCONTRATO   = VT.VTS_CODIGO_CONTRATO
      AND    CN.CNL_CODIGO        = VI_CODIGO_CANAL
      AND    CC.CTC_FECHA_RECISION IS NULL
      AND    VT.VTS_FECHA       BETWEEN VL_FECHA1 AND VL_FECHA2;

      VO_VALOR_COMISION := VL_TOTAL_VTA * (VL_PORC_COM/100);

      RETURN(VO_VALOR_COMISION);
  END VOP_F_CALCULAR_COM_CANAL;

  /*--- FUNCION COMISION X ROTACION ---*/
  FUNCTION VOP_F_CALCULAR_COM_ROT(
        VI_CODIGO_VENDEDOR  IN  CRM_VOP_VENDEDOR.VEN_CODIGO%TYPE,
        VO_VALOR_COMISION   OUT CRM_VOP_VENTA.VTS_VALOR%TYPE)
  RETURN CRM_VOP_VENTA.VTS_VALOR%TYPE AS

        CURSOR CL_VISTA IS SELECT * FROM CRM_VOP_V_ROTACION;

        VL_PORC_COM
CRM_ADM_PARAMETROS.PRM_PORC_COMISION_ROTACION%TYPE;
        VL_TOTAL_VTA   CRM_VOP_VENTA.VTS_VALOR%TYPE;
        VL_TOTAL_VTA_SERV   CRM_VOP_VENTA.VTS_VALOR%TYPE;
        VL_FECHA1      DATE;
        VL_FECHA2      DATE;
        VL_DIA_COMISIONA NUMBER(2);
        VL_DIA_INICIO NUMBER(2);
```

```
 VL_MES_INICIO NUMBER(2);
 VL_MES       VARCHAR2(3);
       VL_ANIO_INICIO NUMBER(4);
       VL_TOTAL_SERVICIOS NUMBER;
       VL_PORC_VENDIDO NUMBER;
  BEGIN

    SELECT PRM_PORC_COMISION_ROTACION
    INTO   VL_PORC_COM
    FROM   CRM_ADM_PARAMETROS;

    SELECT PRM_DIA_COMISIONA
    INTO VL_DIA_COMISIONA
    FROM CRM_ADM_PARAMETROS;

    VL_FECHA2 := TO_DATE(TO_CHAR(VL_DIA_COMISIONA) || '/' ||
TO_CHAR(SYSDATE,'MON') || '/' || TO_CHAR(SYSDATE,'YYYY'));
    --VL_FECHA1 := VL_FECHA2 - 28;
    SELECT TO_NUMBER(TO_CHAR(VL_FECHA2+1,'DD')),
          TO_NUMBER(TO_CHAR(VL_FECHA2,'MM')),
          TO_NUMBER(TO_CHAR(VL_FECHA2,'YYYY'))
    INTO VL_DIA_INICIO,VL_MES_INICIO,VL_ANIO_INICIO
    FROM DUAL;

    IF (VL_MES_INICIO-1 = 0) THEN
      VL_MES_INICIO:=12;
      VL_ANIO_INICIO:=VL_ANIO_INICIO-1;
    ELSE
       VL_MES_INICIO:=VL_MES_INICIO-1;
    END IF;

    VL_MES:=
     CASE
         WHEN VL_MES_INICIO=1 THEN 'JAN'
         WHEN VL_MES_INICIO=2 THEN 'FEB'
         WHEN VL_MES_INICIO=3 THEN 'MAR'
         WHEN VL_MES_INICIO=4 THEN 'APR'
         WHEN VL_MES_INICIO=5 THEN 'MAY'
```

```
      WHEN VL_MES_INICIO=6 THEN 'JUN'
      WHEN VL_MES_INICIO=7 THEN 'JUL'
      WHEN VL_MES_INICIO=8 THEN 'AUG'
      WHEN VL_MES_INICIO=9 THEN 'SEP'
      WHEN VL_MES_INICIO=10 THEN 'OCT'
      WHEN VL_MES_INICIO=11 THEN 'NOV'
      WHEN VL_MES_INICIO=12 THEN 'DEC'
  END;

VL_FECHA1 := TO_CHAR(VL_DIA_INICIO)||'/'||VL_MES||'/'||TO_CHAR(VL_ANIO_INICIO);

SELECT SUM(CANTIDAD) INTO VL_TOTAL_SERVICIOS
FROM CRM_VOP_V_ROTACION;
  VL_TOTAL_VTA:=0;

FOR C IN CL_VISTA LOOP
   VL_PORC_VENDIDO:=ROUND(C.CANTIDAD/VL_TOTAL_SERVICIOS,2)*100;

   IF VL_PORC_VENDIDO>=70 THEN
      SELECT SUM(SC.SEC_PRECIO_SERV) INTO VL_TOTAL_VTA_SERV
      FROM CRM_VOP_VENDEDOR VN, CRM_VOP_CONTRATO_CLIENTE CC,
          CRM_VOP_SERV_CONT SC
      WHERE VN.VEN_CODIGO = VI_CODIGO_VENDEDOR
      AND VN.VEN_CODIGO = CC.CTC_COD_VENDEDOR
      AND CC.CTC_CODCONTRATO = SC.SEC_COD_CONTRATO
      AND CC.CTC_CODCLIENTE = SC.SEC_CODCLIENTE
      AND CC.CTC_FECHA_RECISION IS NULL
      AND (CC.CTC_FEC_INICIO BETWEEN VL_FECHA1 AND VL_FECHA2)
      AND SC.SEC_CODIGO_SERV=C.SRV_CODIGO;

      IF VL_TOTAL_VTA_SERV IS NULL THEN
        VL_TOTAL_VTA_SERV:=0;
      END IF;

      VL_TOTAL_VTA:=VL_TOTAL_VTA+VL_TOTAL_VTA_SERV;
   END IF;
END LOOP;
```

```
    VO_VALOR_COMISION := VL_TOTAL_VTA * (VL_PORC_COM/100);
   RETURN(VO_VALOR_COMISION);
 END VOP_F_CALCULAR_COM_ROT;

 /*--- FUNCION CODIGO CANAL DEL VENDEDOR---*/
 FUNCTION VOP_F_CANAL_VENDEDOR(
        VI_COD_VEND         IN CRM_VOP_VENDEDOR.VEN_CODIGO%TYPE,
        VO_COD_CANAL        OUT CRM_VOP_VENDEDOR.VEN_COD_CANAL%TYPE)
 RETURN CRM_VOP_VENDEDOR.VEN_COD_CANAL%TYPE
 IS
 BEGIN
     SELECT NVL(VE.VEN_COD_CANAL,'000') INTO VO_COD_CANAL
     FROM CRM_VOP_VENDEDOR VE
     WHERE VE.VEN_CODIGO=VI_COD_VEND;

     IF VO_COD_CANAL IS NULL THEN
       VO_COD_CANAL:='';
     END IF;
     RETURN VO_COD_CANAL;
 END VOP_F_CANAL_VENDEDOR;

 /*--- FUNCION COMISION RANGO VENTAS ---*/
 FUNCTION VOP_F_CALCULAR_COM_RV(
        VI_COD_VEND          IN CRM_VOP_VENDEDOR.VEN_CODIGO%TYPE,
        VI_TIPO_CALCULO      IN VARCHAR2,
        VO_COMISION_RV       OUT CRM_VOP_VENDEDOR.VEN_COMISIONES%TYPE)
 RETURN CRM_VOP_VENDEDOR.VEN_COMISIONES%TYPE
 IS
      CURSOR CL_RANGO_VENTAS IS SELECT * FROM CRM_VOP_RANGO_VENTAS;

      VL_PORC_COMISION
CRM_VOP_RANGO_VENTAS.RNV_PORCENTAJE_COMISION%TYPE;
      VL_TOTAL_VTAS_VEND CRM_VOP_VENTA.VTS_VALOR%TYPE;
      VL_FECHA1     DATE;
      VL_FECHA2     DATE;
      VL_DIA_COMISIONA NUMBER(2);
      VL_DIA_INICIO NUMBER(2);
      VL_MES_INICIO NUMBER(2);
```

```
  VL_MES       VARCHAR²)3);
       VL_ANIO_INICIO NUMBER(4);
  BEGIN

     SELECT PRM_DIA_COMISIONA
     INTO VL_DIA_COMISIONA
     FROM CRM_ADM_PARAMETROS;

     --IF VI_TIPO_CALCULO='M' THEN
     IF VI_TIPO_CALCULO┼ =LO┼
_NUMBER(TO_CHAR(VL_FECHA2,'YYYY'))
             INTO VL_DIA_INICIO,VL_MES_INICIO,VL_ANIO_INICIO
        FROM DUAL;

        IF (VL_MES_INICIO-1 = 0) THEN
          VL_MES_INICIO:=12;
          VL_ANIO_INICIO:=VL_ANIO_INICIO-1;
        ELSE
          VL_MES_INICIO:=VL_MES_INICIO-1;
        END IF;
        VL_MES:=
        CASE
           WHEN VL_MES_INICIO=1 THEN 'JAN'
           WHEN VL_MES_INICIO=2 THEN 'FEB'
           WHEN VL_MES_INICIO=3 THEN 'MAR'
           WHEN VL_MES_INICIO=4 THEN 'APR'
           WHEN VL_MES_INICIO=5 THEN 'MAY'
           WHEN VL_MES_INICIO=6 THEN 'JUN'
           WHEN VL_MES_INICIO=7 THEN 'JUL'
           WHEN VL_MES_INICIO=8 THEN 'AUG'
           WHEN VL_MES_INICIO=9 THEN 'SEP'
           WHEN VL_MES_INICIO=10 THEN 'OCT'
           WHEN VL_MES_INICIO=11 THEN 'NOV'
           WHEN VL_MES_INICIO=12 THEN 'DEC'
        END;

        VL_FECHA1 := TO_CHAR(VL_DIA_INICIO)||'/'||VL_MES||'/'||TO_CHAR(VL_ANIO_INICIO);
     END IF;
```

```
    IF VI_TIPO_CALCULO = 'A' THEN
      VL_FECHA2 := TO_DATE(TO_CHAR(VL_DIA_COMISIONA) || '/DEC/' ||
TO_CHAR(SYSDATE,'YYYY'));
      SELECT TO_NUMBER(TO_CHAR(VL_FECHA2+1,'DD')),
          TO_NUMBER(TO_CHAR(VL_FECHA2,'MM'))-11,
          TO_NUMBER(TO_CHAR(VL_FECHA2,'YYYY'))
          INTO VL_DIA_INICIO,VL_MES_INICIO,VL_ANIO_INICIO
      FROM DUAL;

      VL_FECHA1 :=
TO_DATE(TO_CHAR(VL_DIA_INICIO)||'/JAN/'||TO_CHAR(VL_ANIO_INICIO));
    END IF;

    SELECT NVL(SUM(VT.VTS_VALOR),0) INTO VL_TOTAL_VTAS_VEND
    FROM CRM_VOP_VENTA VT
    WHERE (VT.VTS_FECHA BETWEEN VL_FECHA1 AND VL_FECHA2);

    IF VL_TOTAL_VTAS_VEND IS NULL THEN
      VL_TOTAL_VTAS_VEND:=0;
    END IF;

    FOR C IN CL_RANGO_VENTAS LOOP
      IF (VL_TOTAL_VTAS_VEND BETWEEN C.RNV_LIMITE_INFERIOR AND
C.RNV_LIMITE_SUPERIOR) AND C.RNV_TIPO_META=VI_TIPO_CALCULO THEN
        VL_PORC_COMISION:=C.RNV_PORCENTAJE_COMISION;
      ELSE
        VL_PORC_COMISION:=0;
      END IF;
      EXIT WHEN VL_PORC_COMISION <> 0;
    END LOOP;
    VO_COMISION_RV:=VL_TOTAL_VTAS_VEND * (VL_PORC_COMISION/100);

    RETURN VO_COMISION_RV;
  END VOP_F_CALCULAR_COM_RV;

  /*--- FUNCION CALCULAR MULTAS ---*/
  FUNCTION VOP_F_CALCULAR_MULTA(
      VI_CODIGO      IN  CRM_VOP_VENDEDOR.VEN_CODIGO%TYPE,
      VO_VALOR_MULTA     OUT CRM_VOP_VENTA.VTS_VALOR%TYPE)
```

```
  RETURN CRM_VOP_VENTA.VTS_VALOR%TYPE IS
  --FECHA SE RECIBE COMO PARAMETRO
  VL_PORC_MULTA  CRM_ADM_PARAMETROS.PRM_PORC_MULTA%TYPE;
  VL_TOTAL_VTA CRM_VOP_VENTA.VTS_VALOR%TYPE;
  VL_FECHA1      DATE;
  VL_FECHA2      DATE;
  VL_DIA_COMISIONA NUMBER(2);
  VL_DIA_INICIO NUMBER(2);
  VL_MES_INICIO NUMBER(2);
  VL_MES        VARCHAR2(3);
  VL_ANIO_INICIO NUMBER(4);

  BEGIN

      SELECT PRM_PORC_MULTA
      INTO   VL_PORC_MULTA
      FROM   CRM_ADM_PARAMETROS;

      SELECT PRM_DIA_COMISIONA
      INTO VL_DIA_COMISIONA
      FROM CRM_ADM_PARAMETROS;

      VL_FECHA2 := TO_DATE(TO_CHAR(VL_DIA_COMISIONA) || '/' ||
TO_CHAR(SYSDATE,'MON') || '/' || TO_CHAR(SYSDATE,'YYYY'));
      SELECT TO_NUMBER(TO_CHAR(VL_FECHA2+1,'DD')),
           TO_NUMBER(TO_CHAR(VL_FECHA2,'MM')),
           TO_NUMBER(TO_CHAR(VL_FECHA2,'YYYY'))
      INTO VL_DIA_INICIO,VL_MES_INICIO,VL_ANIO_INICIO
      FROM DUAL;

      IF (VL_MES_INICIO-1 = 0) THEN
        VL_MES_INICIO:=12;
        VL_ANIO_INICIO:=VL_ANIO_INICIO-1;
      ELSE
         VL_MES_INICIO:=VL_MES_INICIO-1;
      END IF;
```

```
      VL_MES:=
      CASE
        WHEN VL_MES_INICIO=1 THEN 'JAN'
        WHEN VL_MES_INICIO=2 THEN 'FEB'
        WHEN VL_MES_INICIO=3 THEN 'MAR'
        WHEN VL_MES_INICIO=4 THEN 'APR'
        WHEN VL_MES_INICIO=5 THEN 'MAY'
        WHEN VL_MES_INICIO=6 THEN 'JUN'
        WHEN VL_MES_INICIO=7 THEN 'JUL'
        WHEN VL_MES_INICIO=8 THEN 'AUG'
        WHEN VL_MES_INICIO=9 THEN 'SEP'
        WHEN VL_MES_INICIO=10 THEN 'OCT'
        WHEN VL_MES_INICIO=11 THEN 'NOV'
        WHEN VL_MES_INICIO=12 THEN 'DEC'
      END;

      VL_FECHA1 := TO_CHAR(VL_DIA_INICIO)||'/'||VL_MES||'/'||TO_CHAR(VL_ANIO_INICIO);

      SELECT SUM(VT.VTS_VALOR)
      INTO   VL_TOTAL_VTA
      FROM   CRM_VOP_VENDEDOR VN, CRM_VOP_CONTRATO_CLIENTE CC,
CRM_VOP_VENTA VT
      WHERE  VN.VEN_CODIGO      = VI_CODIGO
      AND    VN.VEN_CODIGO      = CC.CTC_COD_VENDEDOR
      AND    CC.CTC_CODCONTRATO = VT.VTS_CODIGO_CONTRATO
      AND    (CC.CTC_FEC_INICIO  BETWEEN VL_FECHA1 AND VL_FECHA2)
      AND    CC.CTC_FECHA_RECISION IS NOT NULL;

      VO_VALOR_MULTA := VL_PORC_MULTA * (VL_TOTAL_VTA/100);

      RETURN(VO_VALOR_MULTA);
  END VOP_F_CALCULAR_MULTA;

  PROCEDURE VOP_P_INSERTAR_COMISION_MULTA(
    VI_COD_COMISIONES   IN
CRM_VOP_COMISION_MULTA.CMS_COD_COMISIONES%TYPE,
    VI_COD_VENDEDOR     IN CRM_VOP_COMISION_MULTA.CMS_COD_VENDEDOR%TYPE,
    VI_DESCRIPCION      IN CRM_VOP_COMISION_MULTA.CMS_DESCRIPCION%TYPE,
```

```
    VI_VALOR_COMISIONES IN
CRM_VOP_COMISION_MULTA.CMS_VALOR_COMISIONES%TYPE,
    VI_FECHA_COMISIONES IN
CRM_VOP_COMISION_MULTA.CMS_FECHA_COMISIONES%TYPE,
    VI_VALOR_MULTA      IN CRM_VOP_COMISION_MULTA.CMS_VALOR_MULTA%TYPE,
    VI_VALOR_COM_CANAL  IN
CRM_VOP_COMISION_MULTA.CMS_VALOR_COM_CANAL%TYPE,
    VI_VALOR_COM_SERVICIO   IN
CRM_VOP_COMISION_MULTA.CMS_VALOR_COM_SERVICIO%TYPE,
    VI_VALOR_COM_ROTACION   IN
CRM_VOP_COMISION_MULTA.CMS_VALOR_COM_ROTACION%TYPE,
    VI_VALOR_COM_INCENTIVO  IN
CRM_VOP_COMISION_MULTA.CMS_VALOR_COM_INCENTIVO%TYPE
 )
 IS
 BEGIN
     INSERT INTO CRM_VOP_COMISION_MULTA
     (
          CMS_COD_COMISIONES,
          CMS_COD_VENDEDOR,
          CMS_DESCRIPCION,
          CMS_VALOR_COMISIONES,
          CMS_FECHA_COMISIONES,
          CMS_VALOR_MULTA,
          CMS_VALOR_COM_CANAL,
          CMS_VALOR_COM_SERVICIO,
          CMS_VALOR_COM_ROTACION,
          CMS_VALOR_COM_INCENTIVO
     )
     VALUES (
          VI_COD_COMISIONES,
          VI_COD_VENDEDOR,
          VI_DESCRIPCION,
          VI_VALOR_COMISIONES,
          VI_FECHA_COMISIONES,
          VI_VALOR_MULTA,
          VI_VALOR_COM_CANAL,
          VI_VALOR_COM_SERVICIO,
```

```
  VI_VALOR_COM_ROTACION,
        VI_VALOR_COM_INCENTIVO
   );
END VOP_P_INSERTAR_COMISION_MULTA;

/*--- PROCEDIMIENTO PRINCIPAL ---*/
PROCEDURE VOP_P_OBTENER_COMISION(
      VI_COD_VENDEDOR       IN CRM_VOP_VENDEDOR.VEN_CODIGO%TYPE,
      VI_COM_ROTACION       IN VARCHAR2,
      VI_COM_SERVICIO       IN VARCHAR2,
      VI_COM_INCENTIVO      IN VARCHAR2,
      VI_COM_CANAL          IN VARCHAR2,
      VI_COM_RANGO_VTAS     IN VARCHAR2)
IS
     VL_VALOR_MIN_SERV CRM_ADM_PARAMETROS.PRM_VALOR_SERV_MIN_VIP%TYPE;
     VL_COD_CANAL_VEND CRM_VOP_VENDEDOR.VEN_COD_CANAL%TYPE;
     VL_FECHA2     DATE;
     VL_DIA_COMISIONA NUMBER(2);
     VL_CODIGO_CMS    CRM_VOP_COMISION_MULTA.CMS_COD_COMISIONES%TYPE;
     VL_COMISION      CRM_VOP_VENDEDOR.VEN_COMISIONES%TYPE;
     VL_MULTA         CRM_VOP_COMISION_MULTA.CMS_VALOR_MULTA%TYPE;
     VL_COM_SERV      CRM_VOP_VENDEDOR.VEN_COMISIONES%TYPE;
     VL_COM_CANAL     CRM_VOP_VENDEDOR.VEN_COMISIONES%TYPE;
     VL_COM_ROTACION  CRM_VOP_VENDEDOR.VEN_COMISIONES%TYPE;
     VL_COM_RANGO_VTAS CRM_VOP_VENDEDOR.VEN_COMISIONES%TYPE;
     VL_COM_INCENTIVO  CRM_VOP_VENDEDOR.VEN_COMISIONES%TYPE;
BEGIN
   VL_COD_CANAL_VEND:='';
   VL_CODIGO_CMS:='';

   VL_COMISION:=0;
   VL_MULTA:=0;
   VL_COM_SERV:=0;
   VL_COM_CANAL:=0;
   VL_COM_ROTACION:=0;
   VL_COM_RANGO_VTAS:=0;
   VL_COM_INCENTIVO:=0;
```

```
    SELECT MAX(PRM_DIA_COMISIONA) INTO VL_DIA_COMISIONA
    FROM CRM_ADM_PARAMETROS;

    VL_FECHA2 := TO_DATE(TO_CHAR(VL_DIA_COMISIONA) || '/' ||
TO_CHAR(SYSDATE,'MON') || '/' || TO_CHAR(SYSDATE,'YYYY'));

    SELECT PAR.PRM_VALOR_SERV_MIN_VIP INTO VL_VALOR_MIN_SERV
    FROM CRM_ADM_PARAMETROS PAR;

      VL_COD_CANAL_VEND :=
NVL(VOP_F_CANAL_VENDEDOR(VI_COD_VENDEDOR,VL_COD_CANAL_VEND),'000');

      VL_COM_SERV      :=
NVL(VOP_F_CALCULAR_COM_SERV(VI_COD_VENDEDOR,VL_VALOR_MIN_SERV,VL_COM_
SERV),0);
      VL_COM_CANAL     :=
NVL(VOP_F_CALCULAR_COM_CANAL(VI_COD_VENDEDOR,VL_COD_CANAL_VEND,VL_CO
M_CANAL),0);
      VL_COM_ROTACION  :=
NVL(VOP_F_CALCULAR_COM_ROT(VI_COD_VENDEDOR,VL_COM_ROTACION),0);
      VL_COM_RANGO_VTAS :=
NVL(VOP_F_CALCULAR_COM_RV(VI_COD_VENDEDOR,'M',VL_COM_RANGO_VTAS),0);
      VL_COM_INCENTIVO  :=
NVL(VOP_F_CALCULAR_COM_RV(VI_COD_VENDEDOR,'A',VL_COM_RANGO_VTAS),0);

      IF UPPER(VI_COM_SERVICIO)='TRUE' THEN
        VL_COMISION:=VL_COMISION+VL_COM_SERV;
      END IF;

      IF UPPER(VI_COM_CANAL)='TRUE' THEN
        VL_COMISION:=VL_COMISION+VL_COM_CANAL;
      END IF;

      IF UPPER(VI_COM_ROTACION)='TRUE' THEN
        VL_COMISION:=VL_COMISION+VL_COM_ROTACION;
      END IF;
```

```
          IF UPPER(VI_COM_RANGO_VTAS)='TRUE' THEN
            VL_COMISION:=VL_COMISION+VL_COM_RANGO_VTAS;
          END IF;

          IF UPPER(VI_COM_INCENTIVO)='TRUE' THEN
            VL_COMISION:=VL_COMISION+VL_COM_INCENTIVO;
          END IF;

          VL_MULTA := NVL(VOP_F_CALCULAR_MULTA(VI_COD_VENDEDOR,VL_MULTA),0);

        /*--- ACTUALIZAR LA COMISIÓN AL VENDEDOR ---*/
        UPDATE CRM_VOP_VENDEDOR SET VEN_COMISIONES=VL_COMISION-VL_MULTA
        WHERE VEN_CODIGO=VI_COD_VENDEDOR;
        /*--- INSERTAR EN LA TABLA COMISION MULTA ---*/
        VL_CODIGO_CMS:=VOP_F_NUM_SECUENCIAL_CMS(VL_CODIGO_CMS);
        VOP_P_INSERTAR_COMISION_MULTA(
          LPAD(VL_CODIGO_CMS,8,'0'),
          VI_COD_VENDEDOR,
          'COMIS. '||TO_CHAR(SYSDATE,'MON'),
          VL_COMISION,
          VL_FECHA2,
          VL_MULTA,
          VL_COM_CANAL,
          VL_COM_SERV,
          VL_COM_ROTACION,
          VL_COM_INCENTIVO
        );

    END VOP_P_OBTENER_COMISION;

end VOP_K_COMISIONAR;
```

1.8.3.2 Paquete para facturación

```
CREATE OR REPLACE PACKAGE VOP_K_FACTURAR IS
  FUNCTION VOP_F_EXISTE_CLIENTECONTRATO(
        VI_CODIGO   IN  CRM_SCO_CLIENTES.CLI_CODIGO%TYPE,
        ESTADO   OUT BOOLEAN)
  RETURN BOOLEAN;

  FUNCTION VOP_F_NUM_SECUENCIAL_FACTURA(VO_NUMFACTURA OUT
CRM_VOP_CAB_FACTURA.CFAC_NUM_FACTURA%TYPE)
  RETURN CRM_VOP_CAB_FACTURA.CFAC_NUM_FACTURA%TYPE;

  FUNCTION VOP_F_NUM_SECUENCIAL_DETALLE(
        VI_NUMFACTURA IN CRM_VOP_CAB_FACTURA.CFAC_NUM_FACTURA%TYPE,
        VO_SEQDETALLE OUT CRM_VOP_DET_FACTURA.DFAC_NUM_ITEM%TYPE)
  RETURN CRM_VOP_DET_FACTURA.DFAC_NUM_ITEM%TYPE;

  function VOP_F_CALCULAR_IVA(
        VI_SUBTOTAL IN CRM_VOP_CAB_FACTURA.CFAC_SUBTOTAL%TYPE,
        VO_IVA  OUT CRM_VOP_CAB_FACTURA.CFAC_IVA%TYPE)
  return CRM_VOP_CAB_FACTURA.CFAC_IVA%TYPE;

 function VOP_F_CALCULAR_ICE(
        VI_SUBTOTAL IN CRM_VOP_CAB_FACTURA.CFAC_SUBTOTAL%TYPE,
        VO_ICE  OUT CRM_VOP_CAB_FACTURA.CFAC_ICE%TYPE)
  return CRM_VOP_CAB_FACTURA.CFAC_ICE%TYPE;

  function VOP_F_CALCULAR_RECARGO(
        vi_cod_cliente IN CRM_VOP_CAB_FACTURA.CFAC_COD_CLIENTE%TYPE,
        vo_recargo OUT CRM_VOP_CAB_FACTURA.CFAC_RECARGO%TYPE)
  return CRM_VOP_CAB_FACTURA.CFAC_RECARGO%TYPE;
```

```
   FUNCTION VOP_F_CALCULAR_SUBTOTAL_FACT(
         VI_CODCLIENTE IN  CRM_VOP_CONTRATO_CLIENTE.CTC_CODCLIENTE%TYPE,
         VI_COD_CONTRATO IN
CRM_VOP_CONTRATO_CLIENTE.CTC_CODCONTRATO%TYPE,
         VO_SUBTOTAL OUT CRM_VOP_CAB_FACTURA.CFAC_SUBTOTAL%TYPE)
  RETURN CRM_VOP_CAB_FACTURA.CFAC_SUBTOTAL%TYPE;

  FUNCTION VOP_F_CONSTRUYE_FECHA(
       VI_DIA            IN NUMBER,
       VI_INICIO_FIN     IN VARCHAR2,
       VI_MES_HOY        IN NUMBER,
       VI_ANIO_HOY       IN NUMBER,
       VO_FECHA_CONSTRUIDA OUT DATE)
  RETURN DATE;
  FUNCTION VOP_F_CALCULAR_PORRATEADO(
         VI_FECHA_INICIO_CONTRATO IN VARCHAR2,
         VI_FECHA_INICIO_CICLO IN VARCHAR2,
         VI_FECHA_FIN_CICLO IN VARCHAR2,
         VI_TARIFA_BASICA IN NUMBER,
         VO_VALOR_PRORRATEADO OUT NUMBER)
  RETURN NUMBER;

  FUNCTION VOP_F_CONTRATO_CICLO_ANTERIOR(
         VI_COD_CONTRATO IN
CRM_VOP_CONTRATO_CLIENTE.CTC_CODCONTRATO%TYPE,
         VO_RESPUESTA    OUT VARCHAR2)
  RETURN VARCHAR2;

  PROCEDURE VOP_P_FACTURAR_JOB;

  PROCEDURE VOP_P_FACTURAR(
       VI_CODIGO_CLIENTE   IN CRM_SCO_CLIENTES.CLI_CODIGO%TYPE);

END VOP_K_FACTURAR;
/
```

```
CREATE OR REPLACE PACKAGE BODY VOP_K_FACTURAR IS
  -- FUNCTION AND PROCEDURE IMPLEMENTATIONS
  /*--- OBTENER SECUENCIAL DE LA FACTURA ---*/
  FUNCTION VOP_F_NUM_SECUENCIAL_FACTURA(VO_NUMFACTURA OUT
CRM_VOP_CAB_FACTURA.CFAC_NUM_FACTURA%TYPE)
  RETURN CRM_VOP_CAB_FACTURA.CFAC_NUM_FACTURA%TYPE IS

  BEGIN
     VO_NUMFACTURA:=0;
     SELECT MAX(NVL(CFAC_NUM_FACTURA,0)) INTO VO_NUMFACTURA
     FROM CRM_VOP_CAB_FACTURA;
     IF VO_NUMFACTURA IS NULL THEN
       VO_NUMFACTURA:=0;
     END IF;
     VO_NUMFACTURA:=VO_NUMFACTURA+1;
     RETURN VO_NUMFACTURA;
  END;

  /*--- OBTENER SECUENIAL DEL DETALLE ---*/
  FUNCTION VOP_F_NUM_SECUENCIAL_DETALLE(
          VI_NUMFACTURA IN CRM_VOP_CAB_FACTURA.CFAC_NUM_FACTURA%TYPE,
          VO_SEQDETALLE OUT CRM_VOP_DET_FACTURA.DFAC_NUM_ITEM%TYPE)
  RETURN CRM_VOP_DET_FACTURA.DFAC_NUM_ITEM%TYPE IS

  CURSOR C_REGISTROS IS SELECT MAX(NVL(DFAC_NUM_ITEM,0))-- INTO V_SEQDETALLE
                    FROM CRM_VOP_DET_FACTURA
                    WHERE DFAC_NUM_FACTURA=VI_NUMFACTURA;
  BEGIN
     VO_SEQDETALLE:=0;
     OPEN C_REGISTROS;
     LOOP
       FETCH C_REGISTROS INTO VO_SEQDETALLE;
       IF C_REGISTROS%ROWCOUNT=0 THEN
          VO_SEQDETALLE:=0;
       END IF;
       EXIT WHEN C_REGISTROS%NOTFOUND;
END LOOP;
```

```
IF VO_SEQDETALLE IS NULL THEN
      VO_SEQDETALLE:=0;
    END IF;

    VO_SEQDETALLE:=VO_SEQDETALLE+1;
    RETURN TO_CHAR(VO_SEQDETALLE);
 END;

 /*--- EXISTE CONTRATO POR CLIENTE ---*/
 function VOP_F_EXISTE_CLIENTECONTRATO(
        VI_CODIGO   IN  CRM_SCO_CLIENTES.CLI_CODIGO%TYPE,
        ESTADO   OUT BOOLEAN) RETURN BOOLEAN IS

  V_VALOR NUMBER(20);
  V_ESTADO BOOLEAN;
 begin

    SELECT COUNT(*) INTO V_VALOR
    FROM   CRM_VOP_CONTRATO_CLIENTE CO
    WHERE  CO.CTC_CODCLIENTE = VI_CODIGO;

    IF V_VALOR > 0 THEN
      V_ESTADO := TRUE;
    ELSE
      V_ESTADO := FALSE;
    END IF;

    RETURN V_ESTADO;
 end;

 /*--- CALCULAR MONTO IVA ---*/
 function VOP_F_CALCULAR_IVA(
        VI_SUBTOTAL IN CRM_VOP_CAB_FACTURA.CFAC_SUBTOTAL%TYPE,
        VO_IVA  OUT CRM_VOP_CAB_FACTURA.CFAC_IVA%TYPE)
 return CRM_VOP_CAB_FACTURA.CFAC_IVA%TYPE is
      v_porcentaje CRM_ADM_PARAMETROS.PRM_IVA%TYPE;
```

```
  BEGIN
     VO_IVA:=0;
     SELECT NVL(PAR.PRM_IVA,0) INTO V_PORCENTAJE
     FROM CRM_ADM_PARAMETROS PAR;

     IF V_PORCENTAJE IS NULL THEN
       V_PORCENTAJE:=0;
     END IF;
    VO_IVA:=VI_SUBTOTAL*(V_PORCENTAJE/100);

     RETURN VO_IVA;
END;

/*--- CALCULAR MONTO ICE ---*/
FUNCTION VOP_F_CALCULAR_ICE(
       VI_SUBTOTAL IN CRM_VOP_CAB_FACTURA.CFAC_SUBTOTAL%TYPE,
       VO_ICE  OUT CRM_VOP_CAB_FACTURA.CFAC_ICE%TYPE)
RETURN CRM_VOP_CAB_FACTURA.CFAC_ICE%TYPE IS
     V_PORCENTAJE CRM_ADM_PARAMETROS.PRM_ICE%TYPE;
BEGIN
     VO_ICE:=0;
     SELECT NVL(PAR.PRM_ICE,0) INTO V_PORCENTAJE
     FROM CRM_ADM_PARAMETROS PAR;

     IF V_PORCENTAJE IS NULL THEN
       V_PORCENTAJE:=0;
     END IF;

     VO_ICE:=VI_SUBTOTAL*(V_PORCENTAJE/100);
     RETURN VO_ICE;
END;

/*--- CALCULAR RECARGO ---*/
FUNCTION VOP_F_CALCULAR_RECARGO(
     VI_COD_CLIENTE IN CRM_VOP_CAB_FACTURA.CFAC_COD_CLIENTE%TYPE,
     VO_RECARGO OUT CRM_VOP_CAB_FACTURA.CFAC_RECARGO%TYPE)
RETURN CRM_VOP_CAB_FACTURA.CFAC_RECARGO%TYPE IS
```

```
V_total_factura CRM_VOP_CAB_FACTURA.CFAC_TOTAL%TYPE;
      v_total_abonado CRM_VOP_CAB_FACTURA.CFAC_VALOR_ABONO%TYPE;
      v_porcentaje CRM_ADM_PARAMETROS.PRM_INTERES_MORA%TYPE;
   begin
      VO_RECARGO:=0;
      SELECT par.prm_interes_mora INTO v_porcentaje
      FROM crm_adm_parametros par;

      IF V_PORCENTAJE IS NULL THEN
          V_PORCENTAJE:=0;
      END IF;

      SELECT
      SUM(NVL(cfa.cfac_total,0)),SUM(NVL(cfa.cfac_valor_abono,0))
      INTO v_total_factura, v_total_abonado
      FROM CRM_VOP_CAB_FACTURA cfa
      WHERE cfa.cfac_cod_cliente = vi_cod_cliente
      AND cfa.cfac_fecha_vencimiento < sysdate
      AND   cfa.cfac_estado in ('D','A');

      IF V_TOTAL_FACTURA IS NULL THEN
        V_TOTAL_FACTURA:=0;
      END IF;

      IF V_TOTAL_ABONADO IS NULL THEN
        V_TOTAL_ABONADO:=0;
      END IF;
      vo_recargo :=(v_total_factura - v_total_abonado) * v_porcentaje;
      return vo_recargo;

   end;

   /*--- CALCULAR SUBTOTAL FACTURA ---*/
   function VOP_F_CALCULAR_SUBTOTAL_FACT(
          VI_CODCLIENTE IN  CRM_VOP_CONTRATO_CLIENTE.CTC_CODCLIENTE%TYPE,
          VI_COD_CONTRATO IN
CRM_VOP_CONTRATO_CLIENTE.CTC_CODCONTRATO%TYPE,
```

```
VO_SUBTOTAL OUT CRM_VOP_CAB_FACTURA.CFAC_SUBTOTAL%TYPE)
 return CRM_VOP_CAB_FACTURA.CFAC_SUBTOTAL%TYPE is
 begin
      VO_SUBTOTAL:=0;
      SELECT SUM(SC.SEC_PRECIO_SERV) INTO VO_SUBTOTAL
      FROM CRM_VOP_CONTRATO_CLIENTE CC,CRM_VOP_SERV_CONT SC
      WHERE
      CC.CTC_CODCONTRATO=VI_COD_CONTRATO AND
      CC.CTC_CODCLIENTE=VI_CODCLIENTE AND
      CC.CTC_CODCLIENTE=SC.SEC_CODCLIENTE AND
      CC.CTC_CODCONTRATO=SC.SEC_COD_CONTRATO;

      IF VO_SUBTOTAL IS NULL THEN
        VO_SUBTOTAL:=0;
      END IF;

      return VO_SUBTOTAL;
 end;

 /*--- CONSTRUYE FECHA ---*/
 FUNCTION VOP_F_CONSTRUYE_FECHA(
      VI_DIA             IN NUMBER,
      VI_INICIO_FIN      IN VARCHAR2,
      VI_MES_HOY         IN NUMBER,
      VI_ANIO_HOY        IN NUMBER,
      VO_FECHA_CONSTRUIDA OUT DATE)
 RETURN DATE IS
      VL_MES_HOY         NUMBER;
      VL_ANIO_HOY        NUMBER;
      VL_MES             VARCHAR2(3);
      VL_FECHA_CONSTRUIDA DATE;
 BEGIN
    VL_FECHA_CONSTRUIDA:='';
    VL_MES_HOY:=VI_MES_HOY;
    VL_ANIO_HOY:=VI_ANIO_HOY;

    /*SELECT TO_CHAR(SYSDATE,'MM'),TO_CHAR(SYSDATE,'YYYY')
    INTO VL_MES_HOY,VL_ANIO_HOY FROM DUAL;*/
```

```
  IF VI_INICIO_FIN IN ('I') THEN
        VL_MES_HOY:=VL_MES_HOY - 1;
        IF VL_MES_HOY=0 THEN
          VL_MES_HOY:=12;
          VL_ANIO_HOY:=VL_ANIO_HOY-1;
        END IF;
      END IF;

      VL_MES:=
      CASE
          WHEN VL_MES_HOY=1 THEN 'JAN'
          WHEN VL_MES_HOY=2 THEN 'FEB'
          WHEN VL_MES_HOY=3 THEN 'MAR'
          WHEN VL_MES_HOY=4 THEN 'APR'
          WHEN VL_MES_HOY=5 THEN 'MAY'
          WHEN VL_MES_HOY=6 THEN 'JUN'
          WHEN VL_MES_HOY=7 THEN 'JUL'
          WHEN VL_MES_HOY=8 THEN 'AUG'
          WHEN VL_MES_HOY=9 THEN 'SEP'
          WHEN VL_MES_HOY=10 THEN 'OCT'
          WHEN VL_MES_HOY=11 THEN 'NOV'
          WHEN VL_MES_HOY=12 THEN 'DEC'
      END;
      --DBMS_OUTPUT.PUT_LINE(VL_MES_HOY);


VL_FECHA_CONSTRUIDA:=TO_DATE(TO_CHAR(VI_DIA)||'/'||VL_MES||'/'||TO_CHAR(VL_ANIO_
HOY));
      VO_FECHA_CONSTRUIDA:=VL_FECHA_CONSTRUIDA;
      RETURN VO_FECHA_CONSTRUIDA;
  END;

  /*--- FUNCION CALCULAR PRORRATEADO ---*/
  FUNCTION VOP_F_CALCULAR_PORRATEADO(
          VI_FECHA_INICIO_CONTRATO IN VARCHAR2,
          VI_FECHA_INICIO_CICLO    IN VARCHAR2,
          VI_FECHA_FIN_CICLO       IN VARCHAR2,
          VI_TARIFA_BASICA         IN NUMBER,
```

```
   VO_VALOR_PRORRATEADO     OUT NUMBER)
  RETURN NUMBER IS
  VL_VALOR_PRORRATEADO     NUMBER;
  VL_NUM_DIAS_CICLO        NUMBER;
  VL_NUM_DIAS_RESTANTES    NUMBER;
  VL_FECHA_INICIO_CONTRATO DATE;
  VL_FECHA_INICIO_CICLO    DATE;
  VL_FECHA_FIN_CICLO       DATE;
  BEGIN
      VL_VALOR_PRORRATEADO:=0;
      VL_NUM_DIAS_RESTANTES:=0;
      VL_FECHA_INICIO_CONTRATO:=TO_DATE(VI_FECHA_INICIO_CONTRATO);
      VL_FECHA_INICIO_CICLO:=TO_DATE(VI_FECHA_INICIO_CICLO);
      VL_FECHA_FIN_CICLO:=TO_DATE(VI_FECHA_FIN_CICLO);

      IF VL_FECHA_INICIO_CONTRATO BETWEEN VL_FECHA_INICIO_CICLO AND
VL_FECHA_FIN_CICLO THEN
         VL_NUM_DIAS_CICLO:=VL_FECHA_FIN_CICLO - VL_FECHA_INICIO_CICLO;
         DBMS_OUTPUT.PUT_LINE(VL_NUM_DIAS_CICLO);
         VL_NUM_DIAS_RESTANTES:=VL_FECHA_FIN_CICLO - VL_FECHA_INICIO_CONTRATO;
         DBMS_OUTPUT.PUT_LINE(VL_NUM_DIAS_RESTANTES);
         VL_VALOR_PRORRATEADO:=VI_TARIFA_BASICA *
ROUND((VL_NUM_DIAS_RESTANTES / VL_NUM_DIAS_CICLO),4);
      END IF;

      VO_VALOR_PRORRATEADO:=ROUND(VL_VALOR_PRORRATEADO,2);
      RETURN VO_VALOR_PRORRATEADO;
  END;

  /*--- FUNCION CONTRATO CICLO ANTERIOR ---*/
  FUNCTION VOP_F_CONTRATO_CICLO_ANTERIOR(
         VI_COD_CONTRATO IN
CRM_VOP_CONTRATO_CLIENTE.CTC_CODCONTRATO%TYPE,
         VO_RESPUESTA    OUT VARCHAR2)
  RETURN VARCHAR2 IS
        VL_RESPUESTA VARCHAR2(2);
        VL_DIA_INICIO_CICLO NUMBER;
        VL_DIA_FIN_CICLO    NUMBER;
```

```
        VL_MES_CHAR        VARCHAR2(2);
      VL_ANIO_CHAR       VARCHAR2(4);
       VL_MES             NUMBER;
      VL_ANIO            NUMBER;
      VL_MES_ANT         NUMBER;
      VL_ANIO_ANT        NUMBER;
      VL_FECHA_INICIO_CICLO_ANT DATE;
      VL_FECHA_FIN_CICLO_ANT    DATE;
      VL_FECHA_INICIO_CICLO_ACT DATE;
      VL_FECHA_FIN_CICLO_ACT    DATE;
      VL_FECHA_CONTRATO         DATE;
BEGIN
    VL_RESPUESTA:='';

    -- Obtener dia inicio - fin del ciclo
    SELECT CF.CIF_DIA_INICIO_CICLO,CF.CIF_DIA_FIN_CICLO
    INTO VL_DIA_INICIO_CICLO,VL_DIA_FIN_CICLO
    FROM CRM_VOP_CICLO_FACT CF,CRM_VOP_CONTRATO_CLIENTE CC
    WHERE CF.CIF_CODIGO=CC.CTC_CODIGO_CICLO AND
         CC.CTC_CODCONTRATO=VI_COD_CONTRATO;

    -- Obtener fecha de inicio del contrato
    SELECT CC.CTC_FEC_INICIO INTO VL_FECHA_CONTRATO
    FROM CRM_VOP_CONTRATO_CLIENTE CC
    WHERE CC.CTC_CODCONTRATO=VI_COD_CONTRATO;

    -- Obtener fecha inicio - fin ciclo anterior y actual
    SELECT TO_CHAR(SYSDATE,'MM'),TO_CHAR(SYSDATE,'YYYY')
    INTO VL_MES_CHAR,VL_ANIO_CHAR FROM DUAL;

    VL_MES:=TO_NUMBER(VL_MES_CHAR);
    VL_ANIO:=TO_NUMBER(VL_ANIO_CHAR);

    IF (VL_MES-1)=0 THEN
      VL_MES_ANT:=12;
      VL_ANIO_ANT:=VL_ANIO-1;
    ELSE
      VL_MES_ANT:=VL_MES-1;
```

```
        VL_ANIO_ANT:=VL_ANIO;
    END IF;

    VL_FECHA_INICIO_CICLO_ACT :=
VOP_F_CONSTRUYE_FECHA(VL_DIA_INICIO_CICLO,'I',VL_MES,VL_ANIO,VL_FECHA_INICIO_
CICLO_ACT);
    VL_FECHA_FIN_CICLO_ACT :=
VOP_F_CONSTRUYE_FECHA(VL_DIA_FIN_CICLO,'F',VL_MES,VL_ANIO,VL_FECHA_FIN_CICL
O_ACT);

    VL_FECHA_INICIO_CICLO_ANT :=
VOP_F_CONSTRUYE_FECHA(VL_DIA_INICIO_CICLO,'I',VL_MES_ANT,VL_ANIO_ANT,VL_FEC
HA_INICIO_CICLO_ANT);
    VL_FECHA_FIN_CICLO_ANT :=
VOP_F_CONSTRUYE_FECHA(VL_DIA_FIN_CICLO,'F',VL_MES_ANT,VL_ANIO_ANT,VL_FECHA
_FIN_CICLO_ANT);

    -- SI EL CONTRATO SE LO FIRMO EN EL CICLO ANTERIOR --
    IF VL_FECHA_CONTRATO BETWEEN VL_FECHA_INICIO_CICLO_ANT AND
VL_FECHA_FIN_CICLO_ANT THEN
      VL_RESPUESTA:='PR'; -- EL CONTRATO SE PRORRATEA Y SE LO FACTURA
    ELSE
      -- SI EL CONTRATO SE LO FIRMO EN EL PRESENTE CICLO --
      IF VL_FECHA_CONTRATO BETWEEN VL_FECHA_INICIO_CICLO_ACT AND
VL_FECHA_FIN_CICLO_ACT THEN
        VL_RESPUESTA:='NO'; -- EL CONTRATO NO DEBE FACTURARSE (SE LO FIRMÓ EN
EL PRESENTE CICLO)
      ELSE
        VL_RESPUESTA:='SI'; -- EL CONTRATO DEBE FACTURARSE (NORMALMENTE)
      END IF;
    END IF;

    VO_RESPUESTA:=VL_RESPUESTA;
    RETURN VO_RESPUESTA;
  END;
```

```
 /*--- PROCEDIMIENTO PRINCIPAL PARA JOBS ---*/
 procedure VOP_P_FACTURAR_JOB
 IS
  CURSOR CU_CLIENTES IS
  SELECT
       CLI_CODIGO,
       a.CLI_ESTADO ESTADO,
       (b.CLN_NOMBRES  || ' ' || b.CLN_APELLIDOS) AS APELLIDOS
  FROM CRM.CRM_SCO_CLIENTES a,
      CRM.CRM_SCO_CLIENTE_NATURAL b
  WHERE a.CLI_CODIGO=b.CLN_ID AND a.CLI_ESTADO='A'
  UNION
  SELECT
       CLI_CODIGO,
       a.CLI_ESTADO ESTADO,
       (b.CLJ_NOMB_COM  || ' ' || b.CLJ_RAZON) AS APELLIDOS
  FROM CRM.CRM_SCO_CLIENTES a,
      CRM.CRM_SCO_CLIENTE_JURIDICO b
  WHERE a.CLI_CODIGO=b.CLJ_ID AND a.CLI_ESTADO='A';

 BEGIN
    FOR D IN CU_CLIENTES LOOP
      VOP_P_FACTURAR(D.CLI_CODIGO);
    END LOOP;
 END;

 /*--- PROCEDIMIENTO PRINCIPAL ---*/
 procedure VOP_P_FACTURAR(
       VI_CODIGO_CLIENTE   IN CRM_SCO_CLIENTES.CLI_CODIGO%TYPE)
 is
       /* CURSOR PARA OBTENER LOS CONTRATOS VIGENTES TANTO EN LA DURACIÓN
DEL CONTRATO
         COMO PARA SUS SERVICIOS EN ÉL, SIEMPRE Y CUANDO QUE LOS SERVICIOS NO
SE LE
         HAYAN DADO DE BAJA NI HAYAN RESCINDIDO SU CONTRATO */
       CURSOR CU_CONTRATO IS
       SELECT
           CO.CTC_CODCLIENTE,
```

```
  CO.CTC_CODCONTRATO,
           CO.CTC_FEC_INICIO,
           IT.ITE_CODIGO,
           SC.SEC_PRECIO_SERV
      FROM CRM_SCO_CLIENTES CL, CRM_VOP_CONTRATO_CLIENTE CO,
CRM_VOP_SERV_CONT SC, CRM_MKO_ITEMS IT
      WHERE CL.CLI_CODIGO = CO.CTC_CODCLIENTE AND
          CO.CTC_CODCONTRATO = SC.SEC_COD_CONTRATO AND
          CO.CTC_CODCLIENTE  = SC.SEC_CODCLIENTE AND
          SC.SEC_CODIGO_SERV = IT.ITE_CODIGO AND
          CL.CLI_CODIGO = VI_CODIGO_CLIENTE AND
          (SYSDATE BETWEEN CO.CTC_FEC_INICIO AND CO.CTC_FEC_CADUCIDAD) AND
          (SYSDATE BETWEEN SC.SEC_FECHA_ACTIVACION AND SC.SEC_FECHA_FINAL)
AND
          CO.CTC_FECHA_RECISION IS NULL AND
          SC.SEC_COD_TIPO_BAJA IS NULL
      ORDER BY CO.CTC_CODCONTRATO;

      V_ANT_CODCONT CRM_VOP_CONTRATO_CLIENTE.CTC_CODCONTRATO%TYPE;
      V_NUM_FACTURA    CRM_VOP_CAB_FACTURA.CFAC_NUM_FACTURA%TYPE;
      V_NUM_DETALLE    CRM_VOP_DET_FACTURA.DFAC_NUM_ITEM%TYPE;
      V_SUBTOTAL       CRM_VOP_CAB_FACTURA.CFAC_SUBTOTAL%TYPE;
      V_MONIVA         CRM_VOP_CAB_FACTURA.CFAC_IVA%TYPE;
      V_MONICE         CRM_VOP_CAB_FACTURA.CFAC_ICE%TYPE;
      V_RECARGO        CRM_VOP_CAB_FACTURA.CFAC_RECARGO%TYPE;
      V_TOTAL          CRM_VOP_CAB_FACTURA.CFAC_TOTAL%TYPE;
      V_OBSERVACION    CRM_VOP_CAB_FACTURA.CFAC_OBSERVACION%TYPE;
      --VL_VALOR_PRORRATEADO  NUMBER;
      VL_DIA_INICIO_CICLO   NUMBER;
      VL_DIA_FIN_CICLO      NUMBER;
      VL_MES                NUMBER;
      VL_ANIO               NUMBER;
      VL_FECHA_INICIO_CICLO DATE;
      VL_FECHA_FIN_CICLO    DATE;
      VL_RESPUESTA          VARCHAR2(2);
```

```
 begin
   V_ANT_CODCONT:=0;
   V_SUBTOTAL:=0;
   V_MONIVA:=0;
   V_MONICE:=0;
   V_RECARGO:=0;
   V_TOTAL:=0;
   --VL_VALOR_PRORRATEADO:=0;

   FOR C IN CU_CONTRATO LOOP
      /* --- Verificar si este contrato pertenece al ciclo indicado --- */

VL_RESPUESTA:=VOP_F_CONTRATO_CICLO_ANTERIOR(C.CTC_CODCONTRATO,VL_RESP
UESTA);
      IF VL_RESPUESTA<>'NO' THEN
         -- SI EL CONTRATO SE PRORRATEA, SE CALCULA SU VALOR Y SE LO PONE COMO
VALOR DE RECARGO
         IF VL_RESPUESTA='PR' THEN

V_SUBTOTAL:=NVL(VOP_F_CALCULAR_SUBTOTAL_FACT(C.CTC_CODCLIENTE,C.CTC_COD
CONTRATO,V_SUBTOTAL),0);
            SELECT TO_CHAR(SYSDATE,'MM'),TO_CHAR(SYSDATE,'YYYY')
            INTO VL_MES,VL_ANIO FROM DUAL;
            -- Obtener dia inicio - fin del ciclo
            SELECT CF.CIF_DIA_INICIO_CICLO,CF.CIF_DIA_FIN_CICLO
            INTO VL_DIA_INICIO_CICLO,VL_DIA_FIN_CICLO
            FROM CRM_VOP_CICLO_FACT CF,CRM_VOP_CONTRATO_CLIENTE CC
            WHERE CF.CIF_CODIGO=CC.CTC_CODIGO_CICLO AND
                  CC.CTC_CODCONTRATO=C.CTC_CODCONTRATO;

            VL_FECHA_INICIO_CICLO :=
VOP_F_CONSTRUYE_FECHA(VL_DIA_INICIO_CICLO,'I',VL_MES,VL_ANIO,VL_FECHA_INICIO_
CICLO);
            VL_FECHA_FIN_CICLO :=
VOP_F_CONSTRUYE_FECHA(VL_DIA_FIN_CICLO,'F',VL_MES,VL_ANIO,VL_FECHA_FIN_CICL
O);
             V_RECARGO := VOP_F_CALCULAR_PORRATEADO(C.CTC_FEC_INICIO,
                                    VL_FECHA_INICIO_CICLO,
                                    VL_FECHA_FIN_CICLO,
```

```
              V_SUBTOTAL,
              V_RECARGO);
          END IF;

          IF (V_ANT_CODCONT<>C.CTC_CODCONTRATO) THEN
             V_NUM_FACTURA:=VOP_F_NUM_SECUENCIAL_FACTURA(V_NUM_FACTURA);


V_SUBTOTAL:=NVL(VOP_F_CALCULAR_SUBTOTAL_FACT(C.CTC_CODCLIENTE,C.CTC_COD
CONTRATO,V_SUBTOTAL),0);
             V_MONIVA:=NVL(VOP_F_CALCULAR_IVA(V_SUBTOTAL,V_MONIVA),0);
             V_MONICE:=NVL(VOP_F_CALCULAR_ICE(V_SUBTOTAL,V_MONICE),0);
             /*-------------------------------------------------------*/
             V_RECARGO := V_RECARGO +
NVL(VOP_F_CALCULAR_RECARGO(C.CTC_CODCLIENTE,V_RECARGO),0);
             /*-------------------------------------------------------*/
             V_TOTAL:=V_SUBTOTAL+V_MONIVA+V_MONICE+V_RECARGO;

             V_OBSERVACION:='FACT No '||TO_CHAR(V_NUM_FACTURA)||' DEL MES
'||TO_CHAR(SYSDATE,'MM')||'/'||TO_CHAR(SYSDATE,'YYYY');
             -- INSERT CAB
             INSERT INTO CRM_VOP_CAB_FACTURA (
               CFAC_NUM_FACTURA,
               CFAC_COD_CLIENTE,
               CFAC_FECHA_EMISION,
               CFAC_FECHA_VENCIMIENTO,
               CFAC_VALOR_ABONO,
               CFAC_ESTADO,
               CFAC_SUBTOTAL,
               CFAC_IVA,
               CFAC_ICE,
               CFAC_RECARGO,
               CFAC_DESCUENTO,
               CFAC_TOTAL,
               CFAC_OBSERVACION)
             VALUES (
               V_NUM_FACTURA,
               C.CTC_CODCLIENTE,
  SYSDATE,
```

```
                SYSDATE+20,
                0.00,
                'P',
                V_SUBTOTAL,
                V_MONIVA,
                V_MONICE,
                V_RECARGO,
                0.00,
                V_TOTAL,
                V_OBSERVACION
              );
          END IF;
          -- INSERT DET

V_NUM_DETALLE:=NVL(VOP_F_NUM_SECUENCIAL_DETALLE(V_NUM_FACTURA,V_NUM_D
ETALLE),'');
          INSERT INTO CRM_VOP_DET_FACTURA
          (
             DFAC_NUM_FACTURA,
             DFAC_NUM_ITEM,
             DFAC_COD_SERVICIO,
             DFAC_CANTIDAD_SERVICIO,
             DFAC_VALOR_SERVICIO
          )
          VALUES (
             V_NUM_FACTURA,
             V_NUM_DETALLE,
             C.ITE_CODIGO,
             1,
             C.SEC_PRECIO_SERV --V_VALOR_SERVICIO
          );
          V_ANT_CODCONT:=C.CTC_CODCONTRATO;
        END IF;
    END LOOP;
  end;
end VOP_K_FACTURAR;
```

1.8.3.3 Paquete para registrar contratos

```
CREATE OR REPLACE PACKAGE VOP_K_REGISTRA_CONTRATO IS

  -- Author  :
  -- Created : 13/03/2007 22:12:23
  -- Purpose :

  FUNCTION VOP_F_GENERA_CODIGO_VENTA(
           VO_COD_VTA        OUT CRM_VOP_VENTA.VTS_CODIGO_VENTA%TYPE)
  RETURN CRM_VOP_VENTA.VTS_CODIGO_VENTA%TYPE;

  FUNCTION VOP_F_OBTENER_VALOR_VTA(
         VI_COD_CONTRATO     IN
CRM_VOP_CONTRATO_CLIENTE.CTC_CODCONTRATO%TYPE,
         VO_VALOR            OUT CRM_VOP_VENTA.VTS_VALOR%TYPE)
  RETURN CRM_VOP_VENTA.VTS_VALOR%TYPE;

  FUNCTION VOP_F_SEQ_VVS(
         VI_COD_VENTA        IN CRM_VOP_VENT_VEND_SERV.VVS_CODIGO_VENTA%TYPE,
         VO_SECUENCIA        OUT CRM_VOP_VENT_VEND_SERV.VVS_SECUENCIA%TYPE)
  RETURN CRM_VOP_VENT_VEND_SERV.VVS_SECUENCIA%TYPE;

  FUNCTION VOP_F_OBTENER_PRECIO(
        VI_CODIGO_ITEM  IN  CRM_MKO_ITEMS.ITE_CODIGO%TYPE,
        VO_PRECIO           OUT CRM_MKO_PRECIOS.PRE_VALOR%TYPE)
  RETURN CRM_VOP_VENTA.VTS_VALOR%TYPE;

  FUNCTION VOP_F_OBTENER_FECHA_MINIMA(
         VI_FECHA_INICIO     IN CRM_VOP_CONTRATO_CLIENTE.CTC_FEC_INICIO%TYPE,
         VO_FECHA_MINIMA     OUT VARCHAR2)
  RETURN VARCHAR2;

  FUNCTION VOP_F_OBTENER_FECHA_CADUCIDAD(
         VI_FECHA_INICIO    IN CRM_VOP_CONTRATO_CLIENTE.CTC_FEC_INICIO%TYPE,
         VO_FECHA_CADUCA    OUT VARCHAR2)
  RETURN VARCHAR2;
```

```
 FUNCTION VOP_F_INSERTAR_CONTRATO(
        VI_CODCLIENTE       IN CRM_VOP_CONTRATO_CLIENTE.CTC_CODCLIENTE%TYPE,
        VI_FECINICIO        IN CRM_VOP_CONTRATO_CLIENTE.CTC_FEC_INICIO%TYPE,
        VI_FEC_CADUCIDAD    IN
CRM_VOP_CONTRATO_CLIENTE.CTC_FEC_CADUCIDAD%TYPE,
        VI_FEC_MIN_CONTRATO IN
CRM_VOP_CONTRATO_CLIENTE.CTC_FECHA_MIN_CONTRATO%TYPE,
        VI_COD_VENDEDOR     IN
CRM_VOP_CONTRATO_CLIENTE.CTC_COD_VENDEDOR%TYPE,
        VI_COD_FORMA_PAGO   IN
CRM_VOP_CONTRATO_CLIENTE.CTC_COD_FORMA_PAGO%TYPE,
        VI_TIPO_CONTRATO    IN
CRM_VOP_CONTRATO_CLIENTE.CTC_TIPO_CONTRATO%TYPE,
        VI_CODIGO_CICLO     IN
CRM_VOP_CONTRATO_CLIENTE.CTC_CODIGO_CICLO%TYPE,
        VO_CODCONTRATO      OUT
CRM_VOP_CONTRATO_CLIENTE.CTC_CODCONTRATO%TYPE)
  RETURN CRM_VOP_CONTRATO_CLIENTE.CTC_CODCONTRATO%TYPE;
    PROCEDURE VOP_P_INSERTAR_SERV_CONT(
            VI_COD_CONTRATO    IN CRM_VOP_SERV_CONT.SEC_COD_CONTRATO%TYPE,
            VI_CODCLIENTE      IN CRM_VOP_SERV_CONT.SEC_CODCLIENTE%TYPE,
            VI_CODIGO_SERV     IN CRM_VOP_SERV_CONT.SEC_CODIGO_SERV%TYPE,
            VI_SEC_FECHA_ACTIVACION IN
CRM_VOP_SERV_CONT.SEC_FECHA_ACTIVACION%TYPE,
            VI_SEC_FECHA_FINAL  IN CRM_VOP_SERV_CONT.SEC_FECHA_FINAL%TYPE,
            VI_PRECIO_SERV     IN CRM_VOP_SERV_CONT.SEC_PRECIO_SERV%TYPE
    );
    PROCEDURE VOP_P_INSERTAR_VENTA(
            VI_CODIGO       IN CRM_VOP_VENTA.VTS_CODIGO_VENTA%TYPE,
            VI_COD_CONTRATO IN CRM_VOP_VENTA.VTS_CODIGO_CONTRATO%TYPE,
            VI_CODIGO_CANAL IN CRM_VOP_VENTA.VTS_CODIGO_CANAL%TYPE,
            VI_CODIGO_FORMA_PAGO IN
CRM_VOP_VENTA.VTS_CODIGO_FORMA_PAGO%TYPE,
            VI_CODIGO_AGENCIA IN CRM_VOP_VENTA.VTS_CODIGO_AGENCIA%TYPE,
            VI_VALOR        IN CRM_VOP_VENTA.VTS_VALOR%TYPE,
            VI_FECHA        IN CRM_VOP_VENTA.VTS_FECHA%TYPE
    );
```

```
PROCEDURE VOP_P_INSERTAR_VENT_VEND_SERV(
        VI_COD_VENTA       IN CRM_VOP_VENT_VEND_SERV.VVS_CODIGO_VENTA%TYPE,
        VI_SECUENCIA       IN CRM_VOP_VENT_VEND_SERV.VVS_SECUENCIA%TYPE,
        VI_CODIGO_SERVICIO  IN
CRM_VOP_VENT_VEND_SERV.VVS_CODIGO_SERVICIO%TYPE
  );
  PROCEDURE VOP_P_REGISTRAR_VENTA(
        VI_COD_CONTRATO IN CRM_VOP_VENTA.VTS_CODIGO_CONTRATO%TYPE,
        VI_CODIGO_CANAL IN CRM_VOP_VENTA.VTS_CODIGO_CANAL%TYPE,
        VI_CODIGO_FORMA_PAGO IN
CRM_VOP_VENTA.VTS_CODIGO_FORMA_PAGO%TYPE,
        VI_CODIGO_AGENCIA IN CRM_VOP_VENTA.VTS_CODIGO_AGENCIA%TYPE,
        VI_FECHA       IN CRM_VOP_VENTA.VTS_FECHA%TYPE
  );
  PROCEDURE VOP_P_ACTUALIZAR_CONTRATO (
        VI_COD_CONTRATO     IN
CRM_VOP_CONTRATO_CLIENTE.CTC_CODCONTRATO%TYPE,
        VI_CODCLIENTE      IN CRM_VOP_CONTRATO_CLIENTE.CTC_CODCLIENTE%TYPE,
        VI_FECINICIO       IN CRM_VOP_CONTRATO_CLIENTE.CTC_FEC_INICIO%TYPE,
        VI_FEC_CADUCIDAD    IN
CRM_VOP_CONTRATO_CLIENTE.CTC_FEC_CADUCIDAD%TYPE,
        VI_FEC_MIN_CONTRATO IN
CRM_VOP_CONTRATO_CLIENTE.CTC_FECHA_MIN_CONTRATO%TYPE,
        VI_FEC_RECISION     IN
CRM_VOP_CONTRATO_CLIENTE.CTC_FECHA_RECISION%TYPE,
        VI_COD_VENDEDOR     IN
CRM_VOP_CONTRATO_CLIENTE.CTC_COD_VENDEDOR%TYPE,
        VI_COD_FORMA_PAGO   IN
CRM_VOP_CONTRATO_CLIENTE.CTC_COD_FORMA_PAGO%TYPE,
        VI_TIPO_CONTRATO    IN
CRM_VOP_CONTRATO_CLIENTE.CTC_TIPO_CONTRATO%TYPE,
        VI_CODIGO_CICLO     IN
CRM_VOP_CONTRATO_CLIENTE.CTC_CODIGO_CICLO%TYPE,
        VI_COD_TIPO_BAJA    IN
CRM_VOP_CONTRATO_CLIENTE.CTC_COD_TIPO_BAJA%TYPE
  );
```

```
PROCEDURE VOP_P_ACTUALIZAR_SERV_CONT(
          VI_COD_CONTRATO    IN CRM_VOP_SERV_CONT.SEC_COD_CONTRATO%TYPE,
          VI_CODCLIENTE      IN CRM_VOP_SERV_CONT.SEC_CODCLIENTE%TYPE,
          VI_CODIGO_SERV     IN CRM_VOP_SERV_CONT.SEC_CODIGO_SERV%TYPE,
          VI_SEC_FECHA_ACTIVACION IN
CRM_VOP_SERV_CONT.SEC_FECHA_ACTIVACION%TYPE,
          VI_SEC_FECHA_FINAL  IN CRM_VOP_SERV_CONT.SEC_FECHA_FINAL%TYPE,
          VI_PRECIO_SERV     IN CRM_VOP_SERV_CONT.SEC_PRECIO_SERV%TYPE,
          VI_ESTADO          IN CRM_VOP_SERV_CONT.SEC_ESTADO%TYPE,
          VI_COD_TIPO_BAJA    IN CRM_VOP_SERV_CONT.SEC_COD_TIPO_BAJA%TYPE
   );
end VOP_K_REGISTRA_CONTRATO;
/

create or replace package body VOP_K_REGISTRA_CONTRATO is
   FUNCTION VOP_F_GENERA_CODIGO_VENTA(
         VO_COD_VTA       OUT CRM_VOP_VENTA.VTS_CODIGO_VENTA%TYPE)
   RETURN CRM_VOP_VENTA.VTS_CODIGO_VENTA%TYPE
   IS
   BEGIN
      VO_COD_VTA:=0;
      SELECT NVL(MAX(VTS_CODIGO_VENTA),0) INTO VO_COD_VTA
      FROM CRM_VOP_VENTA;
      IF VO_COD_VTA IS NULL THEN
        VO_COD_VTA:=0;
      END IF;
      VO_COD_VTA:=VO_COD_VTA+1;
      RETURN VO_COD_VTA;
   END VOP_F_GENERA_CODIGO_VENTA;

   FUNCTION VOP_F_OBTENER_VALOR_VTA(
         VI_COD_CONTRATO     IN
CRM_VOP_CONTRATO_CLIENTE.CTC_CODCONTRATO%TYPE,
         VO_VALOR          OUT CRM_VOP_VENTA.VTS_VALOR%TYPE)
   RETURN CRM_VOP_VENTA.VTS_VALOR%TYPE
   IS
   BEGIN
      VO_VALOR:=0;
```

```
    SELECT NVL(SUM(SC.SEC_PRECIO_SERV),0) INTO VO_VALOR
    FROM CRM_VOP_CONTRATO_CLIENTE CC,CRM_VOP_SERV_CONT SC
    WHERE CC.CTC_CODCONTRATO=SC.SEC_COD_CONTRATO AND
        CC.CTC_CODCLIENTE=SC.SEC_CODCLIENTE;

    IF VO_VALOR IS NULL THEN
      VO_VALOR:=0;
    END IF;

    RETURN VO_VALOR;
END VOP_F_OBTENER_VALOR_VTA;

FUNCTION VOP_F_SEQ_VVS(
     VI_COD_VENTA       IN CRM_VOP_VENT_VEND_SERV.VVS_CODIGO_VENTA%TYPE,
     VO_SECUENCIA       OUT CRM_VOP_VENT_VEND_SERV.VVS_SECUENCIA%TYPE)
RETURN CRM_VOP_VENT_VEND_SERV.VVS_SECUENCIA%TYPE
IS
BEGIN
    VO_SECUENCIA:=0;
    SELECT NVL(MAX(VS.VVS_SECUENCIA),0) INTO VO_SECUENCIA
    FROM CRM_VOP_VENT_VEND_SERV VS
    WHERE VS.VVS_CODIGO_VENTA=VI_COD_VENTA;

    IF VO_SECUENCIA IS NULL THEN
      VO_SECUENCIA:=0;
    END IF;
    VO_SECUENCIA:=VO_SECUENCIA+1;

    RETURN VO_SECUENCIA;
END VOP_F_SEQ_VVS;

FUNCTION VOP_F_OBTENER_PRECIO(
    VI_CODIGO_ITEM  IN  CRM_MKO_ITEMS.ITE_CODIGO%TYPE,
    VO_PRECIO          OUT CRM_MKO_PRECIOS.PRE_VALOR%TYPE)
RETURN CRM_VOP_VENTA.VTS_VALOR%TYPE
IS
     VL_FECHA_VIGENCIA     CRM_MKO_PRECIOS.PRE_FECHAVIGENCIA%TYPE;
```

```
BEGIN
    VO_PRECIO:=0;
    SELECT NVL(P.PRE_VALOR,0),MAX(P.PRE_FECHAVIGENCIA)
    INTO  VO_PRECIO,VL_FECHA_VIGENCIA
    FROM  CRM_MKO_ITEMS I, CRM_MKO_PRECIOS P
    WHERE I.ITE_CODIGO = P.ITE_CODIGO
    AND   I.ITE_CODIGO = VI_CODIGO_ITEM
    GROUP BY NVL(P.PRE_VALOR,0);

    IF VO_PRECIO IS NULL THEN
      VO_PRECIO:=0;
    END IF;
    RETURN VO_PRECIO;
END VOP_F_OBTENER_PRECIO;

FUNCTION VOP_F_OBTENER_FECHA_MINIMA(
        VI_FECHA_INICIO    IN CRM_VOP_CONTRATO_CLIENTE.CTC_FEC_INICIO%TYPE,
        VO_FECHA_MINIMA    OUT VARCHAR2)
RETURN VARCHAR2
IS
  VL_NUM_MIN CRM_ADM_PARAMETROS.PRM_MES_CAMBIO_CONTRATO%TYPE;
  VL_DIA_INICIO NUMBER(2);
  VL_MES_INICIO NUMBER(2);
  VL_MES_MINIMO NUMBER(2);
  VL_ANIO_INICIO NUMBER(4);
BEGIN
    VL_MES_MINIMO:=0;

    SELECT PRM_MES_CAMBIO_CONTRATO
    INTO VL_NUM_MIN
    FROM CRM_ADM_PARAMETROS;

    SELECT TO_NUMBER(TO_CHAR(VI_FECHA_INICIO,'DD')),
         TO_NUMBER(TO_CHAR(VI_FECHA_INICIO,'MM')),
         TO_NUMBER(TO_CHAR(VI_FECHA_INICIO,'YYYY'))
    INTO   VL_DIA_INICIO,
         VL_MES_INICIO,
         VL_ANIO_INICIO
```

```
        FROM DUAL;
         IF (VL_MES_INICIO+VL_NUM_MIN)>12 THEN
          VL_ANIO_INICIO:=VL_ANIO_INICIO+1;
          VL_MES_INICIO:=12-VL_MES_INICIO;
          VL_MES_MINIMO:=VL_NUM_MIN-VL_MES_INICIO;
        ELSE
          VL_MES_MINIMO:=VL_MES_INICIO+VL_NUM_MIN;
        END IF;

        IF VL_MES_MINIMO=2 AND VL_DIA_INICIO>28 THEN
          VL_DIA_INICIO:=28;
        END IF;

        VO_FECHA_MINIMA:=TO_CHAR(VL_ANIO_INICIO)||'-
'||LPAD(TO_CHAR(VL_MES_MINIMO),2,'0')||'-'||LPAD(TO_CHAR(VL_DIA_INICIO),2,'0');
        RETURN VO_FECHA_MINIMA;
    END VOP_F_OBTENER_FECHA_MINIMA;

    FUNCTION VOP_F_OBTENER_FECHA_CADUCIDAD(
           VI_FECHA_INICIO    IN CRM_VOP_CONTRATO_CLIENTE.CTC_FEC_INICIO%TYPE,
           VO_FECHA_CADUCA    OUT VARCHAR2)
    RETURN VARCHAR2
    IS
      VL_NUM_CADUCA CRM_ADM_PARAMETROS.PRM_MESES_CADUCA_SERVICIO%TYPE;
      VL_DIA_INICIO NUMBER(2);
      VL_MES_INICIO NUMBER(2);
      VL_MES_CADUCA NUMBER(2);
      VL_ANIO_INICIO NUMBER(4);
    BEGIN
        VL_MES_CADUCA:=0;

        SELECT PRM_MESES_CADUCA_SERVICIO
        INTO VL_NUM_CADUCA
        FROM CRM_ADM_PARAMETROS;

        SELECT TO_NUMBER(TO_CHAR(VI_FECHA_INICIO,'DD')),
             TO_NUMBER(TO_CHAR(VI_FECHA_INICIO,'MM')),
             TO_NUMBER(TO_CHAR(VI_FECHA_INICIO,'YYYY'))
        INTO   VL_DIA_INICIO,
```

```
       VL_MES_INICIO,
            VL_ANIO_INICIO
       FROM DUAL;

       IF (VL_MES_INICIO+VL_NUM_CADUCA)>12 THEN
         VL_ANIO_INICIO:=VL_ANIO_INICIO+1;
         VL_MES_INICIO:=12-VL_MES_INICIO;
         VL_MES_CADUCA:=VL_NUM_CADUCA-VL_MES_INICIO;
       ELSE
         VL_MES_CADUCA:=VL_MES_INICIO+VL_NUM_CADUCA;
       END IF;
       IF VL_MES_CADUCA=2 AND VL_DIA_INICIO>28 THEN
         VL_DIA_INICIO:=28;
       END IF;

       VO_FECHA_CADUCA:=TO_CHAR(VL_ANIO_INICIO)||'-
'||LPAD(TO_CHAR(VL_MES_CADUCA),2,'0')||'-'||LPAD(TO_CHAR(VL_DIA_INICIO),2,'0');
       RETURN VO_FECHA_CADUCA;
   END VOP_F_OBTENER_FECHA_CADUCIDAD;

   FUNCTION VOP_F_INSERTAR_CONTRATO(
        VI_CODCLIENTE      IN CRM_VOP_CONTRATO_CLIENTE.CTC_CODCLIENTE%TYPE,
        VI_FECINICIO       IN CRM_VOP_CONTRATO_CLIENTE.CTC_FEC_INICIO%TYPE,
        VI_FEC_CADUCIDAD   IN
CRM_VOP_CONTRATO_CLIENTE.CTC_FEC_CADUCIDAD%TYPE,
        VI_FEC_MIN_CONTRATO IN
CRM_VOP_CONTRATO_CLIENTE.CTC_FECHA_MIN_CONTRATO%TYPE,
        VI_COD_VENDEDOR    IN
CRM_VOP_CONTRATO_CLIENTE.CTC_COD_VENDEDOR%TYPE,
        VI_COD_FORMA_PAGO  IN
CRM_VOP_CONTRATO_CLIENTE.CTC_COD_FORMA_PAGO%TYPE,
        VI_TIPO_CONTRATO   IN
CRM_VOP_CONTRATO_CLIENTE.CTC_TIPO_CONTRATO%TYPE,
        VI_CODIGO_CICLO    IN
CRM_VOP_CONTRATO_CLIENTE.CTC_CODIGO_CICLO%TYPE,
        VO_CODCONTRATO     OUT
CRM_VOP_CONTRATO_CLIENTE.CTC_CODCONTRATO%TYPE
   )RETURN CRM_VOP_CONTRATO_CLIENTE.CTC_CODCONTRATO%TYPE
```

```
IS
 VL_CODCONTRATO      NUMBER;
 BEGIN
    SELECT MAX(CTC_CODCONTRATO) INTO VL_CODCONTRATO
    FROM CRM_VOP_CONTRATO_CLIENTE;

    IF VL_CODCONTRATO IS NULL THEN
      VL_CODCONTRATO:=0;
    END IF;

    VL_CODCONTRATO:=VL_CODCONTRATO+1;

    INSERT INTO CRM_VOP_CONTRATO_CLIENTE
    (
        CTC_CODCONTRATO,
        CTC_CODCLIENTE,
        CTC_FEC_INICIO,
        CTC_FEC_CADUCIDAD,
        CTC_FECHA_MIN_CONTRATO,
        CTC_COD_VENDEDOR,
        CTC_COD_FORMA_PAGO,
        CTC_TIPO_CONTRATO,
        CTC_CODIGO_CICLO
    )
    VALUES (
        VL_CODCONTRATO,
        VI_CODCLIENTE,
        VI_FECINICIO,
        VI_FEC_CADUCIDAD,
        VI_FEC_MIN_CONTRATO,
        VI_COD_VENDEDOR,
        VI_COD_FORMA_PAGO,
        VI_TIPO_CONTRATO,
        VI_CODIGO_CICLO
    );
    VO_CODCONTRATO:=VL_CODCONTRATO;
    RETURN VO_CODCONTRATO;
 END VOP_F_INSERTAR_CONTRATO;
```

```
 PROCEDURE VOP_P_INSERTAR_SERV_CONT(
          VI_COD_CONTRATO    IN CRM_VOP_SERV_CONT.SEC_COD_CONTRATO%TYPE,
          VI_CODCLIENTE      IN CRM_VOP_SERV_CONT.SEC_CODCLIENTE%TYPE,
          VI_CODIGO_SERV     IN CRM_VOP_SERV_CONT.SEC_CODIGO_SERV%TYPE,
          VI_SEC_FECHA_ACTIVACION IN
CRM_VOP_SERV_CONT.SEC_FECHA_ACTIVACION%TYPE,
          VI_SEC_FECHA_FINAL  IN CRM_VOP_SERV_CONT.SEC_FECHA_FINAL%TYPE,
          VI_PRECIO_SERV     IN CRM_VOP_SERV_CONT.SEC_PRECIO_SERV%TYPE
   )
   IS
   VL_PRECIO_SERV   CRM_VOP_SERV_CONT.SEC_PRECIO_SERV%TYPE;
   BEGIN
      VL_PRECIO_SERV:=ROUND(VI_PRECIO_SERV,2);
      INSERT INTO CRM_VOP_SERV_CONT
      (
           SEC_COD_CONTRATO,
           SEC_CODCLIENTE,
           SEC_CODIGO_SERV,
           SEC_FECHA_ACTIVACION,
           SEC_FECHA_FINAL,
           SEC_PRECIO_SERV,
           SEC_ESTADO
       )
       VALUES (
            VI_COD_CONTRATO,
            VI_CODCLIENTE,
            VI_CODIGO_SERV,
            VI_SEC_FECHA_ACTIVACION,
            VI_SEC_FECHA_FINAL,
            VL_PRECIO_SERV,
            'A'
       );
   END VOP_P_INSERTAR_SERV_CONT;
PROCEDURE VOP_P_INSERTAR_VENTA(
          VI_CODIGO       IN CRM_VOP_VENTA.VTS_CODIGO_VENTA%TYPE,
          VI_COD_CONTRATO IN CRM_VOP_VENTA.VTS_CODIGO_CONTRATO%TYPE,
```

```
        VI_CODIGO_CANAL IN CRM_VOP_VENTA.VTS_CODIGO_CANAL%TYPE,
        VI_CODIGO_FORMA_PAGO IN
CRM_VOP_VENTA.VTS_CODIGO_FORMA_PAGO%TYPE,
        VI_CODIGO_AGENCIA IN CRM_VOP_VENTA.VTS_CODIGO_AGENCIA%TYPE,
        VI_VALOR       IN CRM_VOP_VENTA.VTS_VALOR%TYPE,
        VI_FECHA       IN CRM_VOP_VENTA.VTS_FECHA%TYPE
  )
  IS
  BEGIN
     INSERT INTO CRM_VOP_VENTA
     (
          VTS_CODIGO_VENTA,
          VTS_CODIGO_CONTRATO,
          VTS_CODIGO_CANAL,
          VTS_CODIGO_FORMA_PAGO,
          VTS_CODIGO_AGENCIA,
          VTS_VALOR,
          VTS_FECHA
     )
     VALUES (
          VI_CODIGO,
          VI_COD_CONTRATO,
          VI_CODIGO_CANAL,
          VI_CODIGO_FORMA_PAGO,
          VI_CODIGO_AGENCIA,
          VI_VALOR,
          VI_FECHA
     );
  END VOP_P_INSERTAR_VENTA;

  PROCEDURE VOP_P_INSERTAR_VENT_VEND_SERV(
        VI_COD_VENTA       IN CRM_VOP_VENT_VEND_SERV.VVS_CODIGO_VENTA%TYPE,
        VI_SECUENCIA       IN CRM_VOP_VENT_VEND_SERV.VVS_SECUENCIA%TYPE,
        VI_CODIGO_SERVICIO  IN
CRM_VOP_VENT_VEND_SERV.VVS_CODIGO_SERVICIO%TYPE
  )
  IS
 BEGIN
```

```
        INSERT INTO CRM_VOP_VENT_VEND_SERV
        (
             VVS_CODIGO_VENTA,
              VVS_SECUENCIA,
             VVS_CODIGO_SERVICIO
        )
        VALUES (
             VI_COD_VENTA,
             VI_SECUENCIA,
             VI_CODIGO_SERVICIO
        );
    END VOP_P_INSERTAR_VENT_VEND_SERV;

    PROCEDURE VOP_P_REGISTRAR_VENTA(
           VI_COD_CONTRATO IN CRM_VOP_VENTA.VTS_CODIGO_CONTRATO%TYPE,
           VI_CODIGO_CANAL IN CRM_VOP_VENTA.VTS_CODIGO_CANAL%TYPE,
           VI_CODIGO_FORMA_PAGO IN
CRM_VOP_VENTA.VTS_CODIGO_FORMA_PAGO%TYPE,
           VI_CODIGO_AGENCIA IN CRM_VOP_VENTA.VTS_CODIGO_AGENCIA%TYPE,
           VI_FECHA       IN CRM_VOP_VENTA.VTS_FECHA%TYPE
    )
    IS

      CURSOR CL_SERV_CONT IS
                    SELECT SC.SEC_COD_CONTRATO,
                         SC.SEC_CODIGO_SERV
                    FROM CRM_VOP_CONTRATO_CLIENTE CC,CRM_VOP_SERV_CONT SC
                    WHERE CC.CTC_CODCONTRATO=SC.SEC_COD_CONTRATO AND
                        CC.CTC_CODCLIENTE=SC.SEC_CODCLIENTE AND
                        CC.CTC_CODCONTRATO=VI_COD_CONTRATO;

      VL_COD_VENTA          CRM_VOP_VENTA.VTS_CODIGO_VENTA%TYPE;
      VL_VALOR              CRM_VOP_VENTA.VTS_VALOR%TYPE;
      VL_SECUENCIA          CRM_VOP_VENT_VEND_SERV.VVS_SECUENCIA%TYPE;
    BEGIN
       VL_COD_VENTA := NVL(VOP_F_GENERA_CODIGO_VENTA(VL_COD_VENTA),0);
       VL_VALOR    :=
NVL(VOP_F_OBTENER_VALOR_VTA(VI_COD_CONTRATO,VL_VALOR),0);
```

```
+VOP_P_INSERTAR_VENTA(
          VL_COD_VENTA,
          VI_COD_CONTRATO,
          VI_CODIGO_CANAL,
          VI_CODIGO_FORMA_PAGO,
          VI_CODIGO_AGENCIA,
          VL_VALOR,
          VI_FECHA
       );
       FOR C IN CL_SERV_CONT LOOP
          VL_SECUENCIA := NVL(VOP_F_SEQ_VVS(VL_COD_VENTA,VL_SECUENCIA),0);
          VOP_P_INSERTAR_VENT_VEND_SERV(
             VL_COD_VENTA,
             VL_SECUENCIA,
             C.SEC_CODIGO_SERV
          );
       END LOOP;
   END VOP_P_REGISTRAR_VENTA;

   PROCEDURE VOP_P_ACTUALIZAR_CONTRATO (
          VI_COD_CONTRATO     IN
CRM_VOP_CONTRATO_CLIENTE.CTC_CODCONTRATO%TYPE,
          VI_CODCLIENTE       IN CRM_VOP_CONTRATO_CLIENTE.CTC_CODCLIENTE%TYPE,
          VI_FECINICIO        IN CRM_VOP_CONTRATO_CLIENTE.CTC_FEC_INICIO%TYPE,
          VI_FEC_CADUCIDAD    IN
CRM_VOP_CONTRATO_CLIENTE.CTC_FEC_CADUCIDAD%TYPE,
          VI_FEC_MIN_CONTRATO IN
CRM_VOP_CONTRATO_CLIENTE.CTC_FECHA_MIN_CONTRATO%TYPE,
          VI_FEC_RECISION     IN
CRM_VOP_CONTRATO_CLIENTE.CTC_FECHA_RECISION%TYPE,
          VI_COD_VENDEDOR     IN
CRM_VOP_CONTRATO_CLIENTE.CTC_COD_VENDEDOR%TYPE,
          VI_COD_FORMA_PAGO   IN
CRM_VOP_CONTRATO_CLIENTE.CTC_COD_FORMA_PAGO%TYPE,
          VI_TIPO_CONTRATO    IN
CRM_VOP_CONTRATO_CLIENTE.CTC_TIPO_CONTRATO%TYPE,
          VI_CODIGO_CICLO     IN
CRM_VOP_CONTRATO_CLIENTE.CTC_CODIGO_CICLO%TYPE,
          VI_COD_TIPO_BAJA    IN
```

```
CRM_VOP_CONTRATO_CLIENTE.CTC_COD_TIPO_BAJA%TYPE
  )
  IS
   VL_FECHA_RECISION CRM_VOP_CONTRATO_CLIENTE.CTC_FECHA_RECISION%TYPE;
   VL_TIPO_BAJA      CRM_VOP_CONTRATO_CLIENTE.CTC_COD_TIPO_BAJA%TYPE;
  BEGIN
     VL_FECHA_RECISION := VI_FEC_RECISION;
     VL_TIPO_BAJA      := VI_COD_TIPO_BAJA;
     IF VL_TIPO_BAJA <= 0 THEN
       VL_TIPO_BAJA := NULL;
     END IF;
     UPDATE CRM_VOP_CONTRATO_CLIENTE SET
       CTC_CODCLIENTE        = VI_CODCLIENTE,
       CTC_FEC_INICIO        = VI_FECINICIO,
       CTC_FEC_CADUCIDAD     = VI_FEC_CADUCIDAD,
       CTC_FECHA_MIN_CONTRATO = VI_FEC_MIN_CONTRATO,
       CTC_FECHA_RECISION    = VL_FECHA_RECISION,
       CTC_COD_VENDEDOR      = VI_COD_VENDEDOR,
       CTC_COD_FORMA_PAGO    = VI_COD_FORMA_PAGO,
       CTC_TIPO_CONTRATO     = VI_TIPO_CONTRATO,
       CTC_CODIGO_CICLO      = VI_CODIGO_CICLO,
       CTC_COD_TIPO_BAJA     = VL_TIPO_BAJA
     WHERE
     CTC_CODCONTRATO = VI_COD_CONTRATO;

  END VOP_P_ACTUALIZAR_CONTRATO;

  PROCEDURE VOP_P_ACTUALIZAR_SERV_CONT(
         VI_COD_CONTRATO    IN CRM_VOP_SERV_CONT.SEC_COD_CONTRATO%TYPE,
         VI_CODCLIENTE      IN CRM_VOP_SERV_CONT.SEC_CODCLIENTE%TYPE,
         VI_CODIGO_SERV     IN CRM_VOP_SERV_CONT.SEC_CODIGO_SERV%TYPE,
         VI_SEC_FECHA_ACTIVACION IN
CRM_VOP_SERV_CONT.SEC_FECHA_ACTIVACION%TYPE,
         VI_SEC_FECHA_FINAL  IN CRM_VOP_SERV_CONT.SEC_FECHA_FINAL%TYPE,
         VI_PRECIO_SERV      IN CRM_VOP_SERV_CONT.SEC_PRECIO_SERV%TYPE,
         VI_ESTADO          IN CRM_VOP_SERV_CONT.SEC_ESTADO%TYPE,
         VI_COD_TIPO_BAJA    IN CRM_VOP_SERV_CONT.SEC_COD_TIPO_BAJA%TYPE
  )
```

```
IS
        VL_NUM_REGISTROS NUMBER;
        VL_CONTADOR NUMBER;
BEGIN
    SELECT COUNT(*) INTO VL_NUM_REGISTROS
    FROM CRM_VOP_SERV_CONT
    WHERE SEC_COD_CONTRATO = VI_COD_CONTRATO AND
         SEC_CODCLIENTE   = VI_CODCLIENTE   AND
         SEC_CODIGO_SERV  = VI_CODIGO_SERV;

    IF VL_NUM_REGISTROS > 0 THEN
      IF VI_COD_TIPO_BAJA IS NOT NULL OR VI_COD_TIPO_BAJA<>0 THEN
        UPDATE CRM_VOP_SERV_CONT SET
             SEC_CODCLIENTE      = VI_CODCLIENTE,
             SEC_CODIGO_SERV     = VI_CODIGO_SERV,
             SEC_FECHA_ACTIVACION = VI_SEC_FECHA_ACTIVACION,
             SEC_FECHA_FINAL     = VI_SEC_FECHA_FINAL,
             SEC_PRECIO_SERV     = VI_PRECIO_SERV,
             SEC_ESTADO          = VI_ESTADO,
             SEC_COD_TIPO_BAJA   = VI_COD_TIPO_BAJA
        WHERE
            SEC_COD_CONTRATO = VI_COD_CONTRATO AND
            SEC_CODCLIENTE   = VI_CODCLIENTE   AND
            SEC_CODIGO_SERV  = VI_CODIGO_SERV;
      ELSE
        UPDATE CRM_VOP_SERV_CONT SET
             SEC_CODCLIENTE      = VI_CODCLIENTE,
             SEC_CODIGO_SERV     = VI_CODIGO_SERV,
             SEC_FECHA_ACTIVACION = VI_SEC_FECHA_ACTIVACION,
             SEC_FECHA_FINAL     = VI_SEC_FECHA_FINAL,
             SEC_PRECIO_SERV     = VI_PRECIO_SERV,
             SEC_ESTADO          = VI_ESTADO
        WHERE
            SEC_COD_CONTRATO = VI_COD_CONTRATO AND
            SEC_CODCLIENTE   = VI_CODCLIENTE   AND
            SEC_CODIGO_SERV  = VI_CODIGO_SERV;
      END IF;
```

```
        ELSE
        VOP_P_INSERTAR_SERV_CONT(
           VI_COD_CONTRATO,
           VI_CODCLIENTE,
           VI_CODIGO_SERV,
           VI_SEC_FECHA_ACTIVACION,
           VI_SEC_FECHA_FINAL,
           VI_PRECIO_SERV
        );
        END IF;
    END VOP_P_ACTUALIZAR_SERV_CONT;
end VOP_K_REGISTRA_CONTRATO;
/
```

1.9 RowSets utilizados en el módulo de ventas operacionales

En el módulo de Ventas Operacionales, se utilizan Rowsets que están relacionados con DataProviders del mismo nombre. Estos Rowsets contienen sentencias de SQL que permiten obtener registros de toda la tabla o utilizando filtros, e incluso pueden recibir parámetros que son enviados desde la aplicación Java. El resultado de las sentencias SQL es almacenado en los DataProviders para ser manipulados desde el programa.

RowSet: **crm_vop_telefono_vendedorRowSet**
Tabla: **CRM_VOP_TELEFONO_VENDEDOR**
Clase: **mantenimientoVendedor**
mantenimientoTelefonoVendedor
edicionContrato

```
SELECT      a.TLV_CODIGO TLV_CODIGO,
            a.TLV_DESCRIPCION TLV_DESCRIPCION,
            a.TLV_CODIGO_VENDEDOR TLV_CODIGO_VENDEDOR,
            a.TLV_CODIGO_TIPO_TLF TLV_CODIGO_TIPO_TLF,
            c.EMP_APELLIDOS EMP_APELLIDOS,
            d.TPT_DESCRIPCION TPT_DESCRIPCION
FROM        CRM_VOP_TELEFONO_VENDEDOR a,
            CRM_VOP_VENDEDOR b,
            CRM_ADM_EMPLEADOS c,
            CRM_ADM_TIPO_TELEFONO d
WHERE       a.TLV_CODIGO_VENDEDOR=b.VEN_CODIGO
AND         a.TLV_CODIGO_TIPO_TLF=d.TPT_CODIGO
AND         b.VEN_CODIGO=c.EMP_CODIGO
ORDER BY    a.TLV_CODIGO
```

RowSet: **crm_vop_canalRowSet**
Tabla: **CRM_VOP_CANAL**
Clase: **mantenimientoCanal**
mantenimientoVendedor
edicionVisitas
edicionContrato

```
SELECT      CNL_CODIGO,
            CNL_DESCRIPCIONL,
            DECODE(CNL_TIPO_CANAL,'I','Interno','Externo') CNL_TIPO_CANAL,
            CNL_PORC_COMISION_CANAL
FROM CRM.CRM_VOP_CANAL
ORDER BY CNL_CODIGO
```

RowSet: **crm_vop_vendedorRowSet**
Tabla: **CRM_VOP_VENDEDOR**
Clase: **mantenimientoTelefonoVendedor**
mantenimientoVendedor
edicionVisitas
consultaComisiones
consultaVentas

```
SELECT      a.VEN_CODIGO,
            (b.EMP_NOMBRES || '  ' || b.EMP_APELLIDOS ) AS EMP_APELLIDOS,
            b.EMP_NOMBRES,
            b.EMP_COD_DEPARTAMENTO,
            b.EMP_COD_CARGO,
            b.EMP_COD_PROFESION,
            b.EMP_CEDULA,
            b.EMP_SEXO,
            TO_CHAR(b.EMP_FECHA_NACIMIENTO,'DD/MM/YYYY')
            EMP_FECHA_NACIMIENTO,
            b.EMP_ESTADO_CIVIL,
            b.EMP_DOMICILIO,
            b.EMP_TELEFONO,
            b.EMP_CELULAR,
            b.EMP_EMAIL,
            b.EMP_SUELDO,
            b.EMP_JEFE,
            b.EMP_COD_AGENCIA,
            a.VEN_COD_TIPO_VENDEDOR,
            a.VEN_COD_RANGO_VENTAS,
            a.VEN_COD_CANAL,
            a.VEN_COMISIONES
FROM        CRM.CRM_VOP_VENDEDOR a,
            CRM.CRM_ADM_EMPLEADOS b
WHERE       a.VEN_CODIGO=b.EMP_CODIGO AND
            a.VEN_ESTADO<> 'I'ORDER BY a.VEN_CODIGO
```

RowSet: **crm_adm_tipo_telefonoRowSet**
Tabla: **CRM_ADM_TIPO_TELEFONO**
Clase: **mantenimientoTelefonoVendedor**

```
SELECT      a.TPT_CODIGO,
            a.TPT_DESCRIPCION
FROM        CRM.CRM_ADM_TIPO_TELEFONO a
```

RowSet: **crm_sco_tipo_cliente_juridicoRowSet**
Tabla: **CRM_SCO_TIPO_CLIENTE_JURIDICO**
Clase: **edicionProspecto**

```
SELECT      *
FROM        CRM.CRM_SCO_TIPO_CLIENTE_JURIDICO
```

RowSet: **crm_vop_tipo_vendedorRowSet**
Tabla: **CRM_VOP_TIPO_VENDEDOR**
Clase: **MantenimientoTipoVendedor**
mantenimientoVendedor

```
SELECT      *
FROM        CRM.CRM_VOP_TIPO_VENDEDOR
ORDER BY    TIV_COD_TIPO_VENDEDOR
```

RowSet: **crm_vop_rango_ventasRowSet**
Tabla: **CRM_VOP_RANGO_VENTAS**
Clase: **mantenimientoRango**
mantenimientoVendedor

```
SELECT      RNV_COD_RANGO_VENTAS,
            RNV_LIMITE_INFERIOR,
            RNV_LIMITE_SUPERIOR,
            RNV_PORCENTAJE_COMISION, RNV_DESCRIPCION,
            TO_CHAR(RNV_FECHA_INICIAL_RANGO,'DD/MM/YYYY')
            RNV_FECHA_INICIAL_RANGO,
            TO_CHAR(RNV_FECHA_FINAL_RANGO,'DD/MM/YYYY')
            RNV_FECHA_FINAL_RANGO, RNV_TIPO_META
FROM        CRM.CRM_VOP_RANGO_VENTAS
ORDER BY    RNV_COD_RANGO_VENTAS
```

RowSet: **crm_vop_ofertaRowSet**

Tabla: **CRM_VOP_OFERTA**
Clase: **mantenimientoOfertas**

```
SELECT      OFR_COD_OFERTA,
            OFR_DESCRIPCION_OFERTA,
            OFR_MES_INICIAL,
            OFR_MES_FINAL,
            OFR_DESCUENTO,
            OFR_TOTAL
FROM        CRM.CRM_VOP_OFERTA
ORDER BY    OFR_COD_OFERTA
```

RowSet: **crm_vop_ciclo_factRowSet**
Tabla: **CRM_VOP_CICLO_FACT**
Clase. **mantenimientoCicloFact**
edicionContrato

```
SELECT      *
FROM        CRM.CRM_VOP_CICLO_FACT
ORDER BY    CIF_CODIGO
```

RowSet: **crm_adm_empleadosRowSet**
Tabla: **CRM_ADM_EMPLEADOS**
Clase: **mantenimientoVendedor**

```
SELECT      e.EMP_NOMBRES||' '||e.EMP_APELLIDOS as nombres_completos,
            e.*
FROM        CRM_ADM_EMPLEADOS e
```

RowSet: **crm_mko_serviciosRowSet**
Tabla: **CRM_MKO_SERVICIOS**
Clase: **edicionContrato**

```
SELECT *
FROM CRM.CRM_MKO_SERVICIOS
```

RowSet: **crm_adm_estado_civilRowSet**

Tabla: **CRM_ADM_ESTADO_CIVIL**
Clase: **mantenimientoVendedor**
edicionProspecto

```
SELECT *
FROM CRM.CRM_ADM_ESTADO_CIVIL
```

RowSet: **crm_adm_departamentosRowSet**
Tabla: **CRM_ADM_DEPARTAMENTOS**
Clase: **mantenimientoVendedor**

```
SELECT *
 FROM CRM.CRM_ADM_DEPARTAMENTOS
```

RowSet: **crm_sco_clientesRowSet**
Tabla: **CRM_SCO_CLIENTES**
Clase: **edicionVisitas**
edicionSolicitud
edicionContrato
consultaSolicitudes

```
SELECT      CLI_CODIGO,
            a.CLI_ESTADO ESTADO,
            (b.CLN_NOMBRES  || ' ' || b.CLN_APELLIDOS) AS APELLIDOS
FROM        CRM.CRM_SCO_CLIENTES a,
            CRM.CRM_SCO_CLIENTE_NATURAL b
WHERE       a.CLI_CODIGO=b.CLN_ID
AND         a.CLI_ESTADO='A'
UNION
SELECT      CLI_CODIGO,
            a.CLI_ESTADO ESTADO,
            (b.CLJ_NOMB_COM  || ' ' || b.CLJ_RAZON) AS APELLIDOS
FROM        CRM.CRM_SCO_CLIENTES a,
            CRM.CRM_SCO_CLIENTE_JURIDICO b
WHERE       a.CLI_CODIGO=b.CLJ_ID
AND         a.CLI_ESTADO='A'
```

RowSet: **crm_adm_agenciasRowSet**
Tabla: **CRM_ADM_AGENCIAS**
Clase: **mantenimientoVendedor**
edicionContrato
consultaVentas

```
SELECT *
FROM CRM.CRM_ADM_AGENCIAS
```

RowSet: **crm_mko_itemsRowSet1**
Tabla: **CRM_MKO_ITEMS**
Clase: **mantenimientoOfertas**
edicionSolicitud

```
SELECT      I.ITE_CODIGO,
            I.ITE_DESCRIPCION,
            P.PRE_VALOR,
            MAX(P.PRE_FECHAVIGENCIA) PRE_FECHA_VIGENCIA
FROM        CRM.CRM_MKO_ITEMS I,
            CRM_MKO_PRECIOS P
WHERE       I.ITE_CODIGO=P.ITE_CODIGO
            AND         I.ITE_ESTADO='A'
            AND         I.ITE_ESTFACT= 'A'
            AND         P.PRE_ESTADO='A'
            AND         I.SRV_CODIGO=?
GROUP BY    I.ITE_CODIGO,
            I.ITE_DESCRIPCION,
            P.PRE_VALOR;
```

RowSet: **crm_sco_actividadRowSet**
Tabla: **CRM_SCO_ACTIVIDAD**
Clase: **edicionProspecto**

```
SELECT *
FROM CRM.CRM_SCO_ACTIVIDAD
```

RowSet: **crm_vop_visitasRowSet**
Tabla: **CRM_VOP_VISITAS**
Clase: **edicionVisitas**

```
SELECT      V.VST_CONTROL,  V.VST_ID_VISITA,
            ( b.CLN_NOMBRES  || ' ' || b.CLN_APELLIDOS) AS APELLIDOS,
            (E.EMP_NOMBRES || ' ' || E.EMP_APELLIDOS) AS EMP_NOMBRES,
            C.CNL_DESCRIPCIONL,  V.VST_CODIGO_CLIENTE,
            V.VST_CODIGO_VENDEDOR,
            V.VST_CODIGO_CANAL,
            TO_CHAR(V.VST_FECHA_ASIGNACION,'DD/MM/YYYY')
            VST_FECHA_ASIGNACION,
            TO_CHAR(V.VST_FECHA_VISITA,'DD/MM/YYYY')
            VST_FECHA_VISITA,
            DECODE(VST_ESTADO,'V','Visitado','Pendiente') VST_ESTADO,
            V.VST_OBSERVACION
FROM        CRM.CRM_VOP_VISITAS V, CRM.CRM_VOP_VENDEDOR VE,
            CRM.CRM_ADM_EMPLEADOS E, CRM.CRM_VOP_CANAL C,
            CRM.CRM_SCO_CLIENTES a,
            CRM.CRM_SCO_CLIENTE_NATURAL b
WHERE       V.VST_CODIGO_VENDEDOR =  VE.VEN_CODIGO
AND         V.VST_CODIGO_CLIENTE = a.CLI_CODIGO
AND         V.VST_CODIGO_CANAL= C.CNL_CODIGO
AND         VE.VEN_CODIGO = E.EMP_CODIGO
AND         a.CLI_CODIGO=b.CLN_ID
UNION
SELECT      V.VST_CONTROL,   V.VST_ID_VISITA,
            ( b.CLJ_NOMB_COM || ' ' ||  b.CLJ_RAZON) AS APELLIDOS,
            (E.EMP_NOMBRES || ' ' || E.EMP_APELLIDOS) AS EMP_NOMBRES,
            C.CNL_DESCRIPCIONL,  V.VST_CODIGO_CLIENTE,
            V.VST_CODIGO_VENDEDOR,  V.VST_CODIGO_CANAL,
            TO_CHAR(V.VST_FECHA_ASIGNACION,'DD/MM/YYYY')
            VST_FECHA_ASIGNACION,
            TO_CHAR(V.VST_FECHA_VISITA,'DD/MM/YYYY')
            VST_FECHA_VISITA,
            DECODE(VST_ESTADO,'V','Visitado','Pendiente') VST_ESTADO,
            V.VST_OBSERVACION
FROM        CRM.CRM_VOP_VISITAS V,
            CRM.CRM_VOP_VENDEDOR VE,
            CRM.CRM_ADM_EMPLEADOS E,
            CRM.CRM_VOP_CANAL C,
            CRM.CRM_SCO_CLIENTES a,
            CRM.CRM_SCO_CLIENTE_JURIDICO b
WHERE       V.VST_CODIGO_VENDEDOR =  VE.VEN_CODIGO
AND         V.VST_CODIGO_CLIENTE = a.CLI_CODIGO
AND         V.VST_CODIGO_CANAL= C.CNL_CODIGO
AND         VE.VEN_CODIGO = E.EMP_CODIGO AND
a.CLI_CODIGO=b.CLJ_ID
ORDER BY    VST_ID_VISITA
```

RowSet: **crm_vop_prospectoRowSet**
Tabla: **CRM_VOP_PROSPECTO**
Clase: **edicionProspectos**

```
SELECT      PRO_CODIGO_PROSPECTO,
            PRO_NOMBRES,
            PRO_APELLIDOS,
            PRO_CEDULA,PRO_PASAPORTE,
            PRO_SEXO,
            PRO_ESTADO_CIVIL,
            E.ETC_DESCRIPCION,
            TO_CHAR(PRO_FEC_NAC,'DD/MM/YYYY') PRO_FEC_NAC,
            PRO_ACTIVIDAD,
            A.ACT_DESCRIPCION,
            PRO_ESTADO,
            PRO_TIPO_PROSPECTO,
            DECODE(PRO_TIPO_PROSPECTO,1,'Natural',2,'Jur\355dico')
            DESC_TIPO_PROSPECTO,          T.TCL_DESCRIPCION,
            PRO_CODIGO_NAC,
            N.NAC_DESCRIPCION,
            P.PRO_TIPO_CLIENTE_JURIDICO,
            P.PRO_REP_LEG
FROM        CRM_SCO_ACTIVIDAD A,
            CRM_SCO_TIPO_CLIENTE T,
            CRM_ADM_NACIONALIDAD N,
            CRM.CRM_VOP_PROSPECTO P
            LEFT OUTER JOIN CRM_ADM_ESTADO_CIVIL E
            ON  (P.PRO_ESTADO_CIVIL = E.ETC_CODIGO)
WHERE       P.PRO_ACTIVIDAD = A.ACT_ID
AND         P.PRO_TIPO_PROSPECTO = T.TCL_ID
AND         P.PRO_CODIGO_NAC = N.NAC_CODIGO
AND         P.PRO_ESTADO = 'A'
ORDER BY    PRO_CODIGO_PROSPECTO
```

RowSet: **crm_mko_itemsRowSet**
Tabla: **CRM_MKO_ITEMS**
Clase: **mantenimientoOfertas**
edicionSolicitud
edicionContrato

```
SELECT *
FROM CRM.CRM_MKO_SERVICIOS
```

RowSet: **crm_sco_clientes2RowSet**
Tabla: **CRM_SCO_CLIENTES**
Clase: **edicionSolicitud**

```
SELECT     a.CLI_CODIGO CODIGO,
b.CLN_NOMBRES  || ' ' || b.CLN_APELLIDOS AS APELLIDOS
FROM CRM.CRM_SCO_CLIENTES a,
CRM.CRM_SCO_CLIENTE_NATURAL b
WHERE a.CLI_CODIGO=b.CLN_ID
AND a.CLI_ESTADO='A'
UNION
SELECT     a.CLI_CODIGO CODIGO,
b.CLJ_NOMB_COM || b.CLJ_RAZON AS APELLIDOS
FROM CRM.CRM_SCO_CLIENTES a,
CRM.CRM_SCO_CLIENTE_JURIDICO b
WHERE a.CLI_CODIGO=b.CLJ_ID
AND a.CLI_ESTADO='A'"
```

RowSet: **crm_vop_det_solicitudRowSet**
Tabla: **CRM_VOP_DET_SOLICITUD**
Clase: **edicionSolicitud**

```
SELECT        DSL_NUM_SOLICITUD,
              DSL_NUM_ITEM,
              DSL_COD_SERVICIO,
              ITE_DESCRIPCION,
              DSL_CANTIDAD_SERVICIO, DSL_VALOR_SERVICIO,
              (DSL_CANTIDAD_SERVICIO*DSL_VALOR_SERVICIO) AS SUBTOTAL
FROM          CRM.CRM_VOP_DET_SOLICITUD D,
              CRM.CRM_MKO_ITEMS I
WHERE         DSL_NUM_ITEM = '??'AND
              DSL_COD_SERVICIO=ITE_CODIGO
ORDER BY      DSL_NUM_SOLICITUD
```

RowSet: **crm_adm_nacionalidadRowSet**
Tabla: **CRM_ADM_NACIONALIDAD**
Clase: **edicionProspecto**

```
SELECT *
FROM CRM.CRM_ADM_NACIONALIDAD
```

RowSet: **crm_adm_cargosRowSet**
Tabla: **CRM_ADM_CARGOS**
Clase: **mantenimientoVendedor**

```
SELECT *
FROM CRM.CRM_ADM_CARGOS
```

RowSet: **crm_vop_comision_multaRowSet**
Tabla: **CRM_VOP_COMISION_MULTA**
Clase: **consultaComisiones**

```
SELECT ALL   CRM.CRM_VOP_COMISION_MULTA.CMS_COD_COMISIONES,
             CRM.CRM_VOP_COMISION_MULTA.CMS_COD_VENDEDOR,
             CRM.CRM_VOP_COMISION_MULTA.CMS_DESCRIPCION,
             CRM.CRM_VOP_COMISION_MULTA.CMS_VALOR_COMISIONES,
             TO_CHAR(CMS_FECHA_COMISIONES,'YYYY-MM-DD')
             CMS_FECHA_COMISIONES,
             CRM.CRM_VOP_COMISION_MULTA.CMS_VALOR_MULTA,
             CRM.CRM_VOP_COMISION_MULTA.CMS_VALOR_COM_CANAL,
             CRM.CRM_VOP_COMISION_MULTA.CMS_VALOR_COM_SERVICIO,
             CRM.CRM_VOP_COMISION_MULTA.CMS_VALOR_COM_ROTACION,
             CRM.CRM_VOP_COMISION_MULTA.CMS_VALOR_COM_INCENTIVO,
             CRM.CRM_VOP_COMISION_MULTA.USUARIO_INGRESO,
             CRM.CRM_VOP_COMISION_MULTA.FECHA_INGRESO,
             CRM.CRM_VOP_COMISION_MULTA.TERMINAL_INGRESO,
             CRM.CRM_VOP_COMISION_MULTA.USUARIO_MODIFICACION,
             CRM.CRM_VOP_COMISION_MULTA.FECHA_MODIFICACION,
             CRM.CRM_VOP_COMISION_MULTA.TERMINAL_MODIFICACION
FROM         CRM.CRM_VOP_COMISION_MULTA,
             CRM.CRM_VOP_VENDEDOR
WHERE        CRM.CRM_VOP_COMISION_MULTA.CMS_COD_VENDEDOR LIKE ?
AND          CMS_FECHA_COMISIONES >= ?
AND          CMS_FECHA_COMISIONES <= ?
AND          CMS_COD_VENDEDOR = VEN_CODIGOAND VEN_ESTADO <> 'I'
```

RowSet: **crm_vop_det_solicitudRowSet1**
Tabla: **CRM_VOP_DET_SOLICITUD**
Clase: **edicionSolicitud**
consultaSolicitudes

```
SELECT      DSL_NUM_SOLICITUD,
            DSL_NUM_ITEM,
            DSL_COD_SERVICIO,
            ITE_DESCRIPCION,
            DSL_CANTIDAD_SERVICIO, DSL_VALOR_SERVICIO,
            (DSL_CANTIDAD_SERVICIO*DSL_VALOR_SERVICIO) AS SUBTOTAL
FROM        CRM.CRM_VOP_DET_SOLICITUD D,
            CRM.CRM_MKO_ITEMS I
WHERE       DSL_NUM_SOLICITUD = ?
AND         DSL_COD_SERVICIO=ITE_CODIGO");
```

RowSet: **crm_adm_formas_pagoRowSet**
Tabla: **CRM_ADM_FORMAS_PAGO**
Clase: **edicionContrato**

```
SELECT      *
FROM        CRM.CRM_ADM_FORMAS_PAGO
```

RowSet: **crm_sco_tipo_clienteRowSet**
Tabla: **CRM_SCO_TIPO_CLIENTE**
Clase: **edicionVisitas**
edicionProspecto

```
SELECT *
FROM CRM.CRM_SCO_TIPO_CLIENTE
```

RowSet: **crm_adm_empleadosRowSet1**
Tabla: **CRM_ADM_EMPLEADOS**
Clase: **mantenimientoTelefonoVendedor**
mantenimientoVendedor

```
SELECT      EMP_CODIGO,
            EMP_NOMBRES||' '||EMP_APELLIDOS AS EMPLEADO,
            EMP_CEDULA,
            EMP_NOMBRES,
            EMP_APELLIDOS,
            TO_CHAR(EMP_FECHA_NACIMIENTO,'DD/MM/YYYY')
            EMP_FECHA_NACIMIENTO,     EMP_ESTADO_CIVIL,
            EMP_DOMICILIO,
            EMP_TELEFONO,
            EMP_CELULAR,
            EMP_SEXO,
            EMP_EMAIL,
            EMP_COD_PROFESION,
            EMP_COD_AGENCIA,
            EMP_COD_DEPARTAMENTO,
            EMP_COD_CARGO,
            EMP_JEFE,
            EMP_SUELDO
FROM        CRM.CRM_ADM_EMPLEADOS
WHERE       EMP_CODIGO
NOT IN      (SELECT VEN_CODIGO FROM CRM.CRM_VOP_VENDEDOR)
```

RowSet: **crm_vop_cab_solicitudRowSet**
Tabla: **CRM_VOP_CAB_SOLICITUD**
Clase: **consultaSolicitudes**

```
SELECT      CSL_NUM_SOLICITUD,
            CSL_COD_CLIENTE,
            ( b.CLN_NOMBRES  || ' ' || b.CLN_APELLIDOS) AS APELLIDOS,
            TO_CHAR(CSL_FECHA_EMISION,'YYYY-MM-DD')
            CSL_FECHA_EMISION,
            CSL_SUBTOTAL_SOLICITUD,
            CSL_IVA,CSL_ICE,
            CSL_TOTAL_SOLICITUD
FROM        CRM.CRM_VOP_CAB_SOLICITUD CS,
            CRM.CRM_SCO_CLIENTES a,
            CRM.CRM_SCO_CLIENTE_NATURAL b
WHERE       CS.CSL_COD_CLIENTE = a.CLI_CODIGO
AND         a.CLI_CODIGO=b.CLN_ID
AND         CSL_NUM_SOLICITUD >= ?
AND         CSL_NUM_SOLICITUD <= ?
AND         CSL_FECHA_EMISION >= ?
AND         CSL_FECHA_EMISION <= ?
AND         CSL_COD_CLIENTE >= ?
AND         CSL_COD_CLIENTE <= ?
UNION
SELECT      CSL_NUM_SOLICITUD,
            CSL_COD_CLIENTE,
            ( b.CLJ_NOMB_COM || ' ' ||  b.CLJ_RAZON) AS APELLIDOS,
            TO_CHAR(CSL_FECHA_EMISION,'YYYY-MM-DD')
            CSL_FECHA_EMISION,
            CSL_SUBTOTAL_SOLICITUD,
            CSL_IVA,
            CSL_ICE,
            CSL_TOTAL_SOLICITUD
FROM        CRM.CRM_VOP_CAB_SOLICITUD CS,
            CRM.CRM_SCO_CLIENTES a,
            CRM.CRM_SCO_CLIENTE_JURIDICO b
WHERE       CS.CSL_COD_CLIENTE = a.CLI_CODIGO
AND         CS.CSL_COD_CLIENTE = a.CLI_CODIGO
AND         a.CLI_CODIGO=b.CLJ_ID
AND         CSL_NUM_SOLICITUD >= ?
AND         CSL_NUM_SOLICITUD <= ?
AND         CSL_FECHA_EMISION >= ?
AND         CSL_FECHA_EMISION <= ?
AND         CSL_COD_CLIENTE >= ?
AND         CSL_COD_CLIENTE <= ?
```

RowSet: **crm_vop_contrato_clienteRowSet**
Tabla: **CRM_VOP_CONTRATO_CLIENTE**
Clase: **edicionContrato**

```
SELECT      a.ctc_codcontrato,
            TO_CHAR(a.ctc_fec_inicio,'DD/MM/YYYY') as ctc_fec_inicio,
            TO_CHAR(a.ctc_fec_caducidad,'DD/MM/YYYY') as ctc_fec_caducidad,
            TO_CHAR(a.ctc_fecha_min_contrato,'DD/MM/YYYY') as
            ctc_fecha_min_contrato,
            TO_CHAR(a.ctc_fecha_recision,'DD/MM/YYYY') as ctc_fecha_recision,
            a.ctc_codcliente, f.cln_nombres||' '||f.cln_apellidos as cliente,
            a.ctc_cod_vendedor,    e.emp_nombres||' '||e.emp_apellidos as vendedor,
            a.ctc_tipo_contrato,  a.ctc_codigo_ciclo,
            d.cif_descripcion,    a.ctc_cod_forma_pago,   g.frp_descripcion
FROM        CRM.CRM_VOP_CONTRATO_CLIENTE a,
            CRM.CRM_VOP_VENDEDOR b, CRM.CRM_SCO_CLIENTES c,
            CRM.CRM_VOP_CICLO_FACT d,  CRM.CRM_ADM_EMPLEADOS e,
            CRM.CRM_SCO_CLIENTE_NATURAL f,
            CRM.CRM_ADM_FORMAS_PAGO g
WHERE       a.ctc_cod_vendedor=b.ven_codigo
and         b.ven_codigo=e.emp_codigo
and         a.ctc_codcliente=c.cli_codigo
and         c.cli_codigo=f.cln_id
and         a.ctc_codigo_ciclo=d.cif_codigo
and         a.ctc_cod_forma_pago=g.frp_codigo
UNION
SELECT      a.ctc_codcontrato,
            TO_CHAR(a.ctc_fec_inicio,'DD/MM/YYYY') as ctc_fec_inicio,
            TO_CHAR(a.ctc_fec_caducidad,'DD/MM/YYYY') as ctc_fec_caducidad,
            TO_CHAR(a.ctc_fecha_min_contrato,'DD/MM/YYYY') as
            ctc_fecha_min_contrato,
            TO_CHAR(a.ctc_fecha_recision,'DD/MM/YYYY') as ctc_fecha_recision,
            a.ctc_codcliente,
            f.clj_razon||' '||f.clj_nomb_com as cliente,  a.ctc_cod_vendedor,
            e.emp_nombres||' '||e.emp_apellidos as vendedor,
            a.ctc_tipo_contrato,  a.ctc_codigo_ciclo,
            d.cif_descripcion,  a.ctc_cod_forma_pago,    g.frp_descripcion
FROM        CRM.CRM_VOP_CONTRATO_CLIENTE a,
            CRM.CRM_VOP_VENDEDOR b,  CRM.CRM_SCO_CLIENTES c,
            CRM.CRM_VOP_CICLO_FACT d, CRM.CRM_ADM_EMPLEADOS e,
            CRM.CRM_SCO_CLIENTE_JURIDICO f,
            CRM.CRM_ADM_FORMAS_PAGO g
WHERE       a.ctc_cod_vendedor=b.ven_codigo
and         b.ven_codigo=e.emp_codigo
and         a.ctc_codcliente=c.cli_codigo
and         c.cli_codigo=f.clj_id
and         a.ctc_codigo_ciclo=d.cif_codigo
and         a.ctc_cod_forma_pago=g.frp_codigo
ORDER BY    ctc_codcontrato
```

RowSet: **crm_vop_serv_contRowSet1**
Tabla: **CRM_VOP_SERV_CONT**
Clase: **edicionContrato**

```
SELECT      SC.SEC_COD_CONTRATO,
            SC.SEC_CODCLIENTE,
            IT.ITE_DESCRIPCION,
            SC.SEC_CODIGO_SERV,
            TO_CHAR(SC.SEC_FECHA_ACTIVACION,'DD/MM/YYYY')
            SEC_FECHA_ACTIVACION,
            TO_CHAR(SC.SEC_FECHA_FINAL,'DD/MM/YYYY')
            SEC_FECHA_FINAL,    SC.SEC_PRECIO_SERV,
            SC.SEC_COD_TIPO_BAJA,
            TB.TBJ_DESCRIPCION,
            SC.SEC_ESTADO
FROM        CRM.CRM_MKO_ITEMS IT,
            CRM.CRM_VOP_SERV_CONT SC
            LEFT OUTER JOIN CRM.CRM_VOP_TIPO_BAJA TB
            ON (SC.SEC_COD_TIPO_BAJA = TB.TBJ_ID)
WHERE       SC.SEC_CODIGO_SERV = IT.ITE_CODIGO
AND         SC.SEC_COD_CONTRATO = ?
```

RowSet: **crm_vop_tipo_bajaRowSet**
Tabla: **CRM_VOP_TIPO_BAJA**
Clase: **edicionContrato**

```
SELECT *
FROM CRM.CRM_VOP_TIPO_BAJA
```

RowSet: **crm_vop_oferta_servicioRowSet**
Tabla: **CRM_VOP_OFERTA_SERVICIO**
Clase: **mantenimientoOfertas**

```
SELECT ALL  OFE.OSR_NUM_ITEM,
            OFE.OSR_COD_OFERTA,
            OFE.OSR_COD_ITEM,
            OFE.OSR_PRECIO_ITEM,
            IT.ITE_DESCRIPCION
FROM        CRM.CRM_VOP_OFERTA_SERVICIO OFE,
            CRM.CRM_MKO_ITEMS IT
WHERE       OFE.OSR_COD_ITEM = IT.ITE_CODIGO
AND         OFE.OSR_COD_OFERTA = ?
```

RowSet: **crm_vop_ventaRowSet**
Tabla: **CRM_VOP_VENTA**
Clase: **consultaVentas**

```
SELECT      VTS_CODIGO_VENTA,
            VTS_CODIGO_CONTRATO,
            VTS_CODIGO_CANAL,
            VTS_CODIGO_FORMA_PAGO,
            VTS_CODIGO_AGENCIA,
            VTS_VALOR,
            TO_CHAR(VTS_FECHA,'YYYY-MM-DD') VTS_FECHA,
            CNL_CODIGO,
            CNL_DESCRIPCIONL,
            FRP_CODIGO,
            FRP_DESCRIPCION,
            AGE_CODIGO,
            AGE_DESCRIPCION,
            VEN_CODIGO,
            CTC_COD_VENDEDOR
FROM        CRM.CRM_VOP_VENTA,
            CRM.CRM_VOP_CANAL,
            CRM.CRM_ADM_FORMAS_PAGO,
            CRM.CRM_ADM_AGENCIAS,
            CRM.CRM_VOP_VENDEDOR,
            CRM.CRM_VOP_CONTRATO_CLIENTE
WHERE       VTS_CODIGO_CANAL = CNL_CODIGO
AND         VTS_CODIGO_FORMA_PAGO = FRP_CODIGO
AND         VTS_CODIGO_AGENCIA = AGE_CODIGO
AND         VEN_CODIGO = CTC_COD_VENDEDOR
AND         CTC_CODCONTRATO = VTS_CODIGO_CONTRATO
AND         VEN_CODIGO LIKE ?
AND         VTS_CODIGO_AGENCIA >=?
AND         VTS_CODIGO_AGENCIA <=?
AND         VTS_FECHA >= ?
AND         VTS_FECHA <= ?
AND         VEN_ESTADO <> 'I'
```

RowSet: **crm_vop_cab_factura**
Tabla: **CRM_VOP_CAB_FACTURA**
Clase: **pagosOnline**

```
SELECT CFAC_NUM_FACTURA,
          CFAC_COD_CLIENTE,
          CFAC_TOTAL,
          CFAC_OBSERVACION,
          CFAC_ESTADO,
          CLI_CODIGO,
          CLN_ID,
          CLN_CEDULA
FROM CRM.CRM_VOP_CAB_FACTURA, CRM.CRM_SCO_CLIENTES,
        CRM.CRM_SCO_CLIENTE_NATURAL
WHERE CFAC_ESTADO <> 'C'
AND CFAC_COD_CLIENTE =  CLI_CODIGO
AND CLI_CODIGO = CLN_ID
AND (CLN_CEDULA = ? OR CLN_PASAPORTE = ?)
UNION
SELECT CFAC_NUM_FACTURA,
          CFAC_COD_CLIENTE,
          CFAC_TOTAL,
          CFAC_OBSERVACION,
          CFAC_ESTADO,
          CLI_CODIGO,
          CLJ_ID,
          CLJ_RUC
FROM CRM.CRM_VOP_CAB_FACTURA, CRM.CRM_SCO_CLIENTES,
        CRM.CRM_SCO_CLIENTE_JURIDICO
WHERE CFAC_ESTADO <> 'C'
AND CFAC_COD_CLIENTE =  CLI_CODIGO
AND CLI_CODIGO = CLJ_ID
AND CLJ_RUC = ?
```